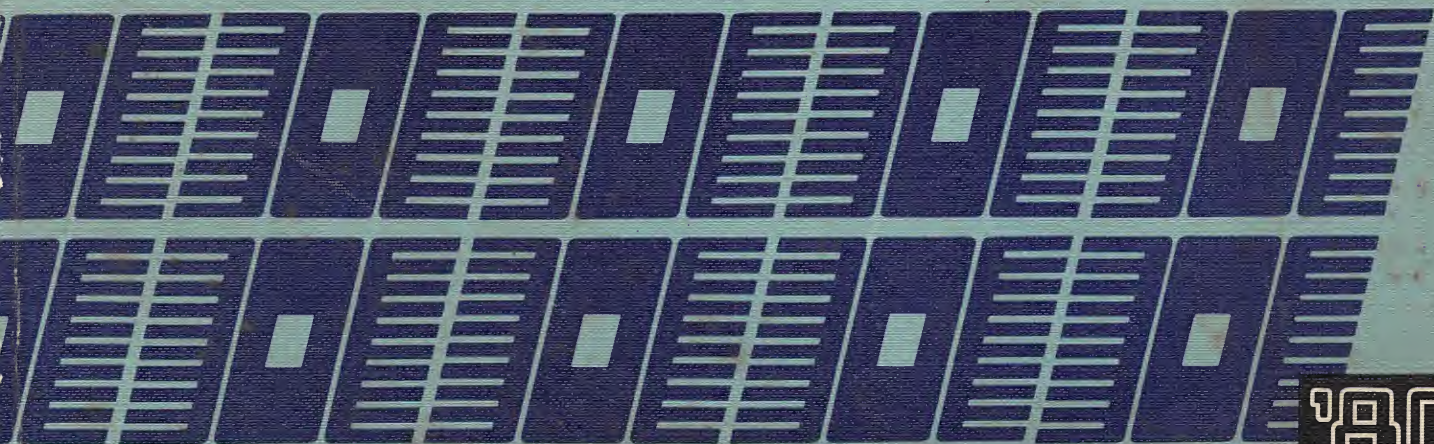


THE COMPARISON OF MICROCOMPUTER "BASIC"

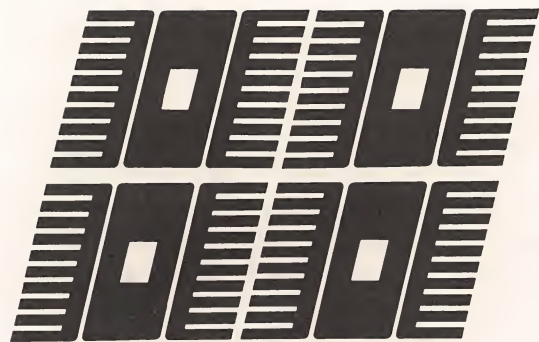
最新 マイコンBASIC規格表

定価 900円



'8Q

CQ出版社



最新 | **マイコンBASIC**
規格表

はじめに

この BASIC 規格表は、パソコンを使う人がそれぞれ 1 冊ずつ手許に置いていれば、とても重宝だと感じていただけるように編集してあります。とかくパソコンのマニュアルは読みにくい、という評判がありますが、この規格表ではプログラム例を多くし、また命令の一般的な書き方よりも具体的な書き方を重視していますので、マニュアルがわりに使うこともできます。またあなたが一つの機種のパソコンに慣れていれば、索引と比較表を駆使することによって、ほかの 29 機種も使うことができます。

年々驚くほどの速度でパソコンの新機種が発表されます。それぞれが異なる BASIC を標準としていることが多いので、使う人はとまどうばかりでしょう。そのような状況の批判から、8 ビット・パソコンの標準 BASIC として MSX が登場してきました。今年から本書にもはじめて収録しましたが、もし MSX がその謳い文句のとおりであるならば、このような規格表は不用となるわけです。しかし現状では、これまで自然発

生的に発達してきた各機種 BASIC のたくさんの機能を MSX はもつに至っておりません。それはこの規格表を見ていただければわかることです。JIS の BASIC もまた同様です。BASIC の良さは、規制から「自由な言語」であったと思うのです。それによってミニフォートランとよばれた時代から、今日のように最も命令数の多い言語にと発達したのだと思います。BASIC は初心者のための言語、ということになっていますが、長年いろいろの言語を使ってきた人が最後に行きつく言語のような気がします。これほど機能が多く、これほど表現を素直にできる言語はほかにありません。コンピュータ言語の専門家達はもっと構造のしっかりした言語を好み、BASIC など馬鹿にする傾向がありますが、言語はあくまでも実用言語でなくてはならず、通常の言語でエスペラント語が力をもたないのが教訓であると思います。

確かに言語はそれぞれに長所をもっています。そしてコンピュータの専門家になろ

うとする人達は、それぞれの言語を学ぶことによってより多くの概念を学べることでしょう。その意味では BASIC のみを学んでいればよい、とは思いません。しかし広く学べば学ぶほど良さがわかる言語、それが BASIC であると思います。どうかこの書を座右において十二分に BASIC を使いこなしていただきたいと思います。

なお編集に当たっては、'83 年版をまとめられた堀内征治、堀内泰輔両先生に御協力をいただき、また CQ 出版社の皆様にも大いにお世話をかけました。引用させていただいたマニュアルなどをご提供いただいた収録各メーカーの方々を含めまして、厚く御礼申し上げます。さらに原稿の整理に尽力を惜しまなかった研究室の学生諸君（浅見・柄沢・中島・丸山・山崎）にも深く感謝します。

最後に、編者のミスから内容が誤っている場合、ご容赦いただくとともに、ご連絡をいただきたく申し添えます。

BASIC 規格表 の 使い方

BASIC 規格表の使い方

この BASIC 規格表には大きく次の二つの使い方があります。

- (1) あなたがすでにひとつの機種に慣れていて、ほかのパソコンをすぐに使いたい場合。
- (2) 新しくパソコンを導入するときの選定の材料として使う場合。

最初の使用法について説明しておきましょう。

例えば、あなたのパソコンが **TRS** であって、画面クリアは **CLS** 命令で行っていたとします。**MZ-80B** に初めて接し、画面クリアはどうしたらよいかというときに、マニュアルで探すのは大変ですね。そんな時、この規格表を手になさってください。

まず巻末のアルファベット順の索引で **CLS** を引き、そのページを開くとそこが画面クリアに関する互換表になっています。互換表は、

- 形式・機能・特徴
- 用語解説（ないページもある）
- プログラム例
- 比較表

の四つの部分から成り立っていますが、その比較表の中の **MZB** の欄の記述、すなわち「**PRINT CHR\$(6)** で画面消去が可能」、により該当する命令が見つかるわけですし。

それでは、**PC-8800** の場合はどうでしょうか。同様に比較表中の **PC88** の欄を見れば OK です。

ところで、その記述中に不明確な用語があった場合、例えば **PC88** の表現中の“テキスト画面”という言葉の意味が不鮮明のときは、もう一度巻末の索引（ほとんどの場合は50音順の索引になるでしょう）を引いて、そこから該当ページを繰り出して下さい。この例では151ページになりますね。そのページの用語欄に、調べたい語の簡単な説明がなされているはずです。これで先程の画面消去のページに戻っていただければ、記述の内容がよく理解できることとします。

なお、さらに詳しい互換表の見方については後で述べます。

さて、あなたが新しくパソコンを導入し

たいとしたら、機種の選定には随分迷われるはずです。価格やデザインも大きな選定要素でしょうが、その機種の BASIC の機能を理解することも重要です。

この規格表の目次には、機能別に全機種の命令を網羅してあります。もちろん、それぞれのパソコンはこれらすべてを備えているわけではありません。そこで、あなたが必要とする命令をこの目次で調べ、該当ページの比較表から、選定の参考資料を引き出すわけです。

例えば、音楽演奏機能を備えている機種が欲しいときには、目次の「音声」に基づいて256ページを開き、それから **MZ** シリーズと **PC3200**、それに **N5200**、**PC-6001**、**SMC-70** などが可能なのだなという情報が得られます。もちろん、ほかの機能の得失についても、合わせて考慮することが必要なのは当然といえます。

以上、本規格表の主な使用法を述べましたが、さらに読者の方々がそれぞれユニークな使用法を試みられることを望んでいます。

✿ 互換表の見方 ✿

互換表は、形式・機能・特徴、用語解説、プログラム例およびほかの機種との比較表より構成されています。この中では、いくつかの約束事に基づいて記述しておりますので、それについて説明しておきましょう。

(1) 形式

従来、各社のマニュアルでの命令またはステートメントの書式は、一般形で書かれていましたが、ここではわかり易くするために、具体的な文章の表現を用いることにしました。ただし、そこで使っている記号には統一した意味があります。表 1 に形式の規則を、表 2 にその記述例を示します。

(2) 用語の解説

機能・特徴および比較表中で使われている用語の中で、とくに説明を要すると思われるものを取りあげました。原則として初めて現れたページに 1 回だけ記述してありますので、そのページに解説がないものは、索引をもとに調べることにします (BASIC 規格表の使い方参照)。

なお、“☐”は参照用語、“☐”は対語になる用語です。

〈表 1〉 形式の規則

引用記号	意味	引用記号	意味
A, A1, A2	単精度の変数	2, 3, 3.14	単精度の定数
B, B1, B2	単精度の変数の配列 (次元は100, 150, 200)	“サトウ”, “スズキ”, “タテ”, “ヨコ”, “アイウエオカキクケコ”	文字定数
I, J, K	単精度の制御変数	“AFILE”, “BFILE”, “afile”	ファイル名
L	論理変数		
M, N	整数変数	#5, #7	入力ファイル
C#, C1#, C2#	倍精度の変数	#6, #8	出力ファイル
D#, D1#, D2#	倍精度の変数の配列 (次元は100, 150, 200)	#9	入出力ファイル
E\$, E1\$, E2\$	文字列変数	┌┐	省略可能
F\$, F1\$, F2\$	文字列変数の配列 (次元は100, 150, 200)	┌┐ *	省略および繰り返し可能
		↑	特に必要なもの
		└┘	その部分の説明
3.14159265358979 2.2362067925910	倍精度の定数	空白	ことわらなければ省略してもよい

〈表 2〉 形式の記述例

	マニュアルなどでの表現	本規格表での表現
例 1	CLOSE┌[(#) <ファイル番号> [, (#) <ファイル番号>...]	CLOSE #5, #6 ↑ 必 要 ┌┐ 要 省略可
例 2	FIELD (#) <ファイル番号>, <フィールド幅> AS <文字変数> ...	FIELD #5, 10 AS E\$, 5 AS E1\$ 省略可 フィールド幅 *省略・繰り返し可 (ただし ┌┐ による説明は省くことがある)
例 3	FOR <変数名> = X TO Y [STEP Z]	FOR I=1 TO 100 STEP 2 省略可

(3) 比較表

ここでは、ディスク BASIC のある機種を主に取りあげました。またその中でも、趣味または業務用のパーソナル・コンピュータに限定して選びました。

機種の順序は、だいたい発売順にそっています。また、いくつものソフトが用意されているものについては、代表的なものだけを取りあげました。

各機種名は、略されて記されています。**表 3**は各機種の正式名称・製作者名・本書で取りあげたソフトウェアの正式名称です。また互換性のあるものは、そのソフトウェア名または機種名をあげました。

比較表中の記述形式はおおむね次の表現に準じています。ただし、ここでの冒頭のアルファベットは機種名を表すこととします。

① AAA ・本文参照

形式・機能・特徴およびプログラム例（これを「本文」という）は、機種 AAA について記述されている。

② BBB ・本文と同じ

場合によって若干のニュアンスの差はあるが、本文とほとんど同じである。

③ CCC ・△△△△△△……

基本的に本文と同じであるが、△△△△△△……に関して差異が認められる（「本

〈表 3〉 機種・ソフトウェア名一覧

略 称	正 式 名 称	社 名*	ソフトウェア名	互 換
APL	APPLE II J-plus	ア ッ プ ル		APPLE IIe
TRS _I	TRS-80 モデル I	タ ン デ ィ	レベル II BASIC	TRS-80 ^{(*)1} モデル III
CPM		マ イ ク ロ ソ フ ト	MICROSOFT™ BASIC-80	CPM CBASIC
M223	M223 mark II	ソ ー ド	E-BASIC (CBASIC)	M203 /M200 各 mark シリーズ /M23/M20
MZK	MZ-80K	シャ ー プ	MZ-80K DISK BASIC5520	MZ-80C /MZ-80K2 /MZ-80K2E
CBM	CBM 4032	コ モ ド ー ル	コ モ ド ー ル BASIC	
PC8	PC-8001	NEC	N-BASIC	PC-8001MK II ^{(*)2}
TRS _{II}	TRS-80 モデル II	タ ン デ ィ	モデル II BASIC	
PC3	PC-3200S	シャ ー プ	SHARP BASIC	PC-3100 ^{(*)3} S
IF8 ₂	IF800 model 20	沖	OKI BASIC	IF800model130 ^{(*)4}
レベル ₃	ベーシック・マスタ レベル ₃ MB-6890	日 立	LEVEL-3 BASIC	MB-6891 マーク 5
C180	PANAFACOM C-180	パ ナ ファ コ ム	リアル・タイム BASIC-B	F9450ターミナル C18（松下）
M243	M243 mark II	ソ ー ド	E-BASIC (CBASIC)	M243 各 mark シリーズ
MZB	MZ-80B	シャ ー プ	MZ-80B DISK BASIC SB6520	
BUB	BUBCOM 80	SFC		
FM8	FUJITSU MICRO8	富 士 通	F-BASIC	FM7
PSP	PASOPIA	東 芝	OA DISK BASIC	
PC88	PC-8801	NEC	N ₈₈ (N) BASIC	PC-9801/E/F ^{(*)6}
N52	N5200 モデル 05	NEC	PTOS BASIC	
PC6	PC-6001	NEC	N ₆₀ BASIC	PC6001MK II

〈表3〉 つづき

略 称	正 式 名 称	社 名*	ソフトウェア名	互 換
MLT	MULTI-16	三 菱	M-BASIC	スタンド・アロン M-BASIC
HC	HC-20	EPSON	EPSON BASIC	
FP11	FP-1100	カ シ オ	C82-BASIC	FP-1000 ^(*4)
SMC	SMC-70	ソ ニ ー	SONY BASIC	
MZ35	MZ-3500	シャープ	MZ-3500 BASIC	
X1Hu	X1	シャープ	Hu BASIC	
MB16	MB-16000	日 立	16000 BASIC	
IBM55	IBMマルチステー ション5550	IBM	IBM BASIC	
PC100	PC-100	NEC	N100-BASIC	PC8801, PC9801 ^(*7)
MSX	MSX BASICに対 応する全機種	各 社	MSX-BASIC	MSX-BASIC 対 応機種

※略称

(*1) ただし、TIME\$ 追加。

(*2) ただし、PC8001MKIIでは、以下の命令を追加。命令はCMDとともに用いる。BLOAD, BSAVE, COPY, CLS, COLOR@, SCREEN, VIEW, CIRCLE, COLOR, GET@, LINE, PAINT, POINT, PRESET, PSET, PUT@、またCMDをつけずに、STATUS POINT, STATUSVIEWがある。

(*3) ただし、以下の命令はPC-3100Sにはない。VRUN, ERASE, SUBEND, POP, MOVE, SEARCH, WAIT, ENTER, LINE, PL, RV, RE, TABLE, SCROL, COLOR, CCHAIN以上。また、拡張ROMをつけると以下の命令は可能。DELETE, REN, CALL, SUB, KEYIN, DEFFN, AND, OR, NOT, STRING\$, MID\$, SPA, CDEG, CDMS, FN以上。

(*4) ただし、グラフィック命令(Gカーソル, SYMBOL, SCALE)追加。

(*5) ただし、SCREEN, GET@, PUT@ 命令はない。

(*6) DRAW 命令が異なる。

(*7) 直接プログラムは走らないが、ユーティリティのファイル・コンバータを用いて相互にプログラムを変換できる。

文と同じ、ただし」が省かれていると解すればよい)。

④ DDD ・CCC と同じ。

同一比較表の中で機種 CCC と同様な表現である。

⑤ EEE ××××××

・△△△△△……………

形式が本文と異なるとき、×××××××がこの機種についての形式を表し、・△△△△△……………で機能を表す。これが省かれた時は、機能・特徴が本文に準ずることを意味する。

⑥ FFF ・該当なし。ただし△△△……

本文の形式に該当するものはないが、ほかの命令で同様の機能を代替えすることができるとき、「該当なし、ただし」の部分を省く場合もある。

⑦ GGG —

本文にある命令が、当該機種にない場合、あるいは動作が確認しきれない場合。

なお、全機種の中で、ある1機種だけにしかない命令は、原則として比較することせず、それに最も近いと思われる命令の比較表中に「関連して……」と表現して添えました。

表4に参照したマニュアル名を掲げます。

〈表4〉 参照マニュアル名

APL	APPLE II SOFT BASIC PROGRAMMING REFERENCE MANUAL	レベル3	・日立 パーソナル コンピュータ ベーシック マスタ レベル3 MB-6890 取扱説明書	HC	・HC-20 BASIC REFERENCE MANUAL ・TF20 HC-20 Disk BASIC REFERENCE MANUAL
TRS _I	TRS-80 LEVEL II BASIC Owner's MANUAL		・HITACHI BASIC MASTER MB-6890 LEVEL 3 DISK BASIC マニュアル	FP11	・FP-1000/1100 C82-BASIC 文法書
CPM	MICROSOFT BASIC-80 REFERENCE MANUAL	C180	・FACOM 9450 パーソナルターミナル BASIC 使用者の手引き ・FACOM 9450 パーソナルターミナル操作手引き	SMC	・Sony BASIC 文法書 Part 1 ・Sony BASIC 文法書 Part 2
M223	SORD MICRO COMPUTER M200 mark SERIES			MZ35	・MZ-3500 BASIC LANGUAGE MANUAL ・MZ-3500 BASIC LANGUAGE MANUAL 付録 ・MZ-3500 OWNER'S MANUAL
MZK	SHARP MZ-80 SERIES DISK BASIC	M243	M243 mark SERIES MDOS/XMS オペレーティング・システム		
CBM	・CBM4032 Personal Computer User's MANUAL ・Commodore BASIC User's REFERENCE MANUAL	MZB	パーソナル コンピュータ MZ-80B DISK BASIC MANUAL	X1Hu	・X1 CZ-802C BASIC MANUAL ・X1 CZ-800C DISK BASIC MANUAL
		BUB	BUBCOM 80 BASIC 文法		
PC8	・PC8001 USER'S MANUAL ・PC8001 N-BASIC REFERENCE MANUAL	FM8	・FM-8 システム解説 ・F-BASIC 活用事典	MB16	・16000 SERIES BASIC MANUAL ・16000 SERIES MS-DOS MANUAL
		PSP	PASOPIA OA-BASIC 言語説明書		
TRS _{II}	・TRS-80 Model II 取扱説明書 ・ディスク・オペレーティング・システム/ディスク・ベーシック	PC88	・PC8801 USER'S MANUAL ・PC8801 N88-BASIC N-BASIC REFERENCE MANUAL	IBM55	・IBMマルチステーション5550 日本語DOSユーザーズ・ガイド ・IBMマルチステーション5550 BASIC インタープリター・ユーザーズ・ガイド
PC3	・SHARP PC-3100(S)取扱説明書 ・SHARP PC-3000シリーズ ミニフロピ・ディスク取扱説明書		N52		
IF8 ₂	・if800 model 20 OKI-BASIC 文法解説書 ・if800 model 20 取扱説明書	PC6	・N60-BASIC ・N60-拡張 BASIC	PC100	・N100-BASIC リファレンス マニュアル
		MLT	・MULTI16 M-BASIC 文法説明書	MSX	HB-55, FM-X 各MSX BASIC 文法書, 他

規格表

目 次

1 基本事項

使用文字	15
? : ; , " ' シンボル	16
へー*／＋－¥ 算術演算子	18
<=> 関係演算子	19
＋＝<> 文字列演算子	20
NOT AND OR XOR IMP EQV 論理演算子	21
整数型定数 数値定数 (その1)	22
単精度型定数 数値定数 (その2)	23
倍精度型定数 数値定数 (その3)	24
文字定数	25
変数	26
ラベル	28
配列	29
型変換	30
スクリーン・エディタの使い方	32
コントロール文字	34

2 プログラムの制御・編集

▶ プログラム制御

RUN プログラムの実行	36
TRON, TROFF トレース文	38
STEP ON, STEP OFF ステップ実行	39
SEARCH 文字列サーチ	40
CHAIN 実行プログラムの変更	41
MERGE プログラムのマージ	42
PLOAD オーバレイ	44
GODBY CLI モードへの復帰	46
BYE CLI モードへの復帰	47

▶ プログラム編集

EDIT 行の修正	48
NEW プログラム領域のクリア	49
AUTO 行番号の自動発生	50
LIST プログラムのリスト	52
LLIST リスト印刷	54
RENUM 行番号の整理	56
DELETE 行の削除	57
NO LIST 機密の保持	58
KEYロック キーの指定と解除	59
LOGIN プログラム・エリアの切り換え	60
PCOPY ほかのプログラム・エリアにプログラムをコピー	61

3 ステートメント (1)

▶ プログラム構成命令

REM 注釈文	63
STOP 停止	64
CONT プログラムの再実行	65
END プログラムの終了	66

▶ データの代入

LET 代入文	67
READ データ文からの入力	68
DATA データ文	69
RESTORE 読み出し位置の指定	70
SWAPVALUE 値の交換	71

▶ 定義・宣言

DEFINT 暗黙の整数型指定	72
-----------------	----

DEFSNG 暗黙の単精度指定	73
DEFDBL 暗黙の倍精度指定	74
DEFSTR 暗黙の文字型指定	75
DEFCHR\$ 表示文字の字形定義	76
DEF SEG ベース・アドレスの設定	77
DIM 配列の定義	78
OPTION BASE 配列下限値の指定	80
CLEAR メモリ・クリア	81
ERASE 変数の削除	83
DEFFN 関数定義文	84
COMMON 変数引き渡し宣言	85
DEFUSR 機械語開始アドレス指定	86
PARACT, PARAEND, URGENCY 並列処理部の定義	87

▶ ループ・分岐

FOR 繰り返し	89
NEXT 繰り返しの端末	90
IF 条件文	91
GOTO 飛び越し	92
ON GOTO 多方向への分岐	93
WHILE ループ条件文	95
UNTIL ループ条件文	96
AT EOF #n ファイル終了の分岐	97

▶ エラー処理

ON ERROR GOTO エラー処理分岐	98
ERROR エラーの模擬発生	99
TRAP ON 入出力エラー分岐指定	100
RESUME エラー処理後の再開	101
ERL エラー発生行	102
ERR エラー処理	103

▶ サブルーチン

GOSUB 副プログラムへの分岐	104
ON GOSUB 複数サブルーチンへの分岐	105

RETURN 副プログラムからのもどり	106
POP 上位ルーチンへのもどり	107
CALL 副プログラムの呼び出し	108
SUB 副プログラムの先頭	109
SUBEND 副プログラムの終了	110
SUBEXIT 副プログラムの出口	111

▶ 機械語

USR 機械語ルーチンの呼び出し	112
PEEK 指定メモリからの直接読み出し	113
POKE 指定メモリへの直接書き込み	114
FRE 未使用バイト数の表示	116
VARPTR 変数の番地表示	118
EXEC 機械語プログラムの実行	119
MSAVE, MLOAD 機械語のセーブとロード	121
MON 機械語モードへの移行	123

④ ステートメント (2) 入出力

▶ 入力命令

INPUT キーボードからの入力	125
LINE INPUT 行入力文	127
INKEY\$ キー入力の検出	129
INPUT\$ 一定文字数の入力	131
INPUT #n シーケンシャル・ファイル入力文	132
LINE INPUT #n シーケンシャル・ファイル行入力文	133
GET ランダム・ファイルからの入力	135
READ #n 内部コードのファイルからの入力	136
INP ポートからの入力	137

▶ 出力命令

PRINT CRT への出力	139
PRINT USING 書式指定プリント (その1)	141
PRINT USING 書式指定プリント (その2)	143
PRINT USING 書式指定プリント (その3)	144

PRINT USING 書式指定プリント (その4)	146
PRINT USING 書式指定プリント (その5)	148
LPRINT プリンタへの出力	149
LPRINT USING 書式指定プリンタ出力	150
COPY 画面のコピー	151
PRINT #n シーケンシャル・ファイル出力文	152
PRINT #n USING 書式指定ファイル出力	153
PUT ランダム・ファイルへの出力	154
LSET フィールド内左詰め書き込み	155
RSET フィールド内右詰め書き込み	156
WRITE #n 内部コードのファイルへの出力	157
OUT ポート出力	158

▶ ディスク入出力・ファイル

LOAD プログラムのロード	159
SAVE プログラムの格納	161
NAME ファイル名変更	163
KILL ファイルの消去	165
SET ファイル属性の設定と解除	167
MOUNT マウント文	169
REMOVE リムーブ文	170
MAXFILE バッファ数設定	171
OPEN _{ファイル} ファイルのオープン文	172
CLOSE ファイル・クローズ文	174
FIELD フィールド文	175
FILES ディレクトリ出力	177
LFILES ディレクトリのプリンタ出力	179
FORMAT フォーマット	180
CREATE ディレクトリの作成	182
FILECOPY ファイルのコピー	184
MARGIN #n レコード長の変更	185
RESTORE #n ファイル・ポインタの移動	186
VSAVE, VLOAD VIDEO-RAMデータのセーブとロード	187
SWAP _{プログラム} プログラム実行の交換	188

▶ ディスク関数

DSKI\$ セクタ入力	189
DSKO\$ セクタ出力	190
DSKF ディスク・フリー	191
EOF ファイルの終わりの検出	192
EOR #n レコード終端コードの指定	193
ATTR\$ ファイルの属性表示	194
FPOS 現在のセクタ番号の表示	196
LOC レコード番号の表示	197
LOF ファイルの大きさの表示	199

▶ カセット入出力

CLOAD カセット・ロード	200
CSAVE カセット・セーブ	202
CLOAD? カセット・ロード・ベリファイ	204
ROPEN カセット・テープのリード・オープン	206
WOPEN カセット・テープのライト・オープン	207
MOTOR カセット・テープ・レコーダのモータ制御	208

▶ 一般画面

CONSOLE 画面行制御	209
ERASE 画面の消去	210
WIDTH 画面の文字数を設定	212
ROLL スクローリングの指示	214
SCREEN _{関数} 表示文字の読み込み	216
LOCATE カーソル位置の設定	217
CSRLIN カーソルの縦位置表示	219
POS カーソルの横位置表示	220
PEN ライト・ペンのエリア指定	221
PEN _{関数} ライト・ペン関数	222
MOUSE マウスの機能設定	224

▶ グラフィックス

SCREEN _{モード} グラフィック画面のモード設定	225
--	-----

VIEW, WINDOW 画面領域設定	227
PMAP 座標の変換	229
COLOR 色指定	230
LINE 線引き (その1)	232
LINE 線引き (その2)	233
LINE 線引き (その3)	234
CIRCLE 楕円の表示	236
PAINT 塗りつぶし	238
PSET ドット・セット	240
PRESET ドット・リセット	242
POINT ドット判定	244
SYMBOL 指定角度, 指定の大きさの文字列を表示	246
GCURSOR グラフィック・カーソル座標の読み取り	247
GET@ 画面のセーブ	248
GET@A 画面のセーブ	250
SPRITE\$ スプライト・パターンの登録	251
PUT@ 配列画面出力	252
PUT@A 配列画面出力	254
PUT SPRITE 変数画面出力	255

▶ 音 声

BEEP ブザー	256
MUSIC 音楽演奏	257
TEMPO テンポの指定	258

▶ 通信回線

OPEN _{入出力} 入出力ポートのオープン文	259
COM (n) ON 通信回線からの割り込み許可	261
ON COM (n) GOSUB 通信回線の入力線の分岐	262
TERM 端末機として使用	263

▶ 割り込み

TIME ON, TIME OFF, TIME STOP タイマ割り込み 許可, 禁止	265
ON TIME GOSUB タイマ割り込み先定義	266

ON KEY GOSUB, OFF KEY キーによる割り込み	267
PEN ON, PEN OFF, PEN STOP ライト・ペンによる 割り込み許可, 禁止	269
ON PEN GOSUB ライト・ペンによる割り込み	270

▶ 入出力, その他

KEY _{ファンクション} ファンクション・キーの定義	271
KEY LIST ファンクション・キーの内容表示	272
KSAVE, KLOAD ファンクション・キーの内容のセーブ, ロード	273
SLEEP 時間監視	274
WAIT _{時間} 時間監視	275
WAIT _{データ} 入力データの監視	276
LPOS プリント・ヘッドの位置表示	277
PAGE 紙送り	278
BUBR, BUBW バブル・メモリのアクセス	279

5 関 数

▶ 数値関数

SQR 平方根	280
SIN, COS, TAN 三角関数	281
ATN 三角関数逆正接	282
ANGLE 角度モードの指定	283
EXP 指数関数	284
LOG ₁₀ 常用対数	285
LOG _e 自然対数	286
ABS 絶対値	287
MOD 剰余	288
SGN 符号関数	289
ROUND 丸め	290
FRAC 小数部の取り出し	291
MAX, MIN 最大値, 最小値	292
RND 乱数の発生	293
RANDOMIZE 乱数の初期設定	294

FAC 階乗	295
SL%, SR% 2進数の左/右シフト	296

▶ 文字列関数

LEN 文字列の長さ	297
RIGHT\$ 右側文字列の取り出し	298
LEFT\$ 左側文字列の取り出し	299
MID\$ 中間文字列の取り出し	300
STRING\$ 1文字の繰り返し	301
SPC 空白の印字	302
INSTR スtringの探索	303
TAB 印字桁の指定	304
TIME\$ 時刻	306
DATE\$ 日付け	308
DAY 曜日	310

▶ 型変換

INT 整数化	311
---------------	-----

CINT 整数型化	312
FIX 整数部分	313
CDBL 単精度の倍精度化	314
CSNG 倍精度の単精度化	315
CVI 内部コードの整数化	316
CVS 内部コードの単精度数値化	317
CVD 内部コードの倍精度数値化	318
MKI\$ 整数値の内部コード化	319
MKS\$ 単精度数値の内部コード化	320
MKD\$ 倍精度数値の内部コード化	321
NORMAL, FIXED, FLOAT 出力形式の指定	322
CHR\$ アスキー・コードの文字化	323
ASC アスキー・コード化	325
VAL 文字列の数値化	326
STR\$ 数値の文字列化	327
HEX\$ 16進への変換	328
OCT\$ 8進数への変換	329
CDMS, CDEG 10進, 60進の変換	330

使用文字

形式：英文字(大文字, 小文字) カナ 数字 (0~9)
 特殊記号 カナ小文字 カナ記号 グラフィック¹⁾
 記号

機能：BASIC で使用できる文字。

用語：¹⁾グラフィック グラフを描く意味。特定のグラフに限定されず、一般的な図形表示はすべてグラフィックである。

²⁾JISコード 日本工業規格(Japanese Industrial Standardの略)によって定められたコード。

プログラム例：キャラクタ・セット(使用文字)を印字する。

```
10 FOR I=&H20 TO &HF7 STEP20
20 FOR J=0 TO 20
30 LPRINT CHR$(I+J);
40 NEXT J
50 LPRINT
60 NEXT I
70 END
```

実行例

```
!"#$%&'()*+,-./01234
456789:;<=>?@ABCDEFGHIJ
KLMNOPQRSTUVWXYZ[\]^_
`abcdefghijklmnopqrstuvwxyz
{|}~
■□△▽
「」・ラアイウエオ
ヤヨッアイウエオカククコサシセソ
タチツテトナニヌネノヒフヘホマミムメモ
ヤヨラリルレロワン°=≡≡▲▼◆
◆◆◆◆◆○/△円年月日時分秒
```

参照：PRINT, REM, シンボル

APL ・ CPM と同じ。

TRS_I ・ 本文と同じ。

CPM ・ カナ関係およびグラフィック記号は不可。

M223 ・ 本文と同じ。

MZK ・ 英小文字は不可。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 本文と同じ。

IF8₂ ・ 本文と同じ。

レベル₃ ・ ひらがなの使用も可。

C180 ・ 本文と同じ。

M243 ・ 外国文字とひらがなが付加。また簡易グラフィック文字も付加。

MZB ・ MZK と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ ひらがな、ギリシャ文字、ロシア文字の使用可。

PC88 ・ 本文と同じ。

N52 ・ 英文字は大文字のみ。
 ・ 漢字(JIS²⁾ 第一水準の漢字と非漢字)の使用可。
 ・ グラフィック記号の使用は不可。

PC6 ・ レベル₃と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。

SMC ・ レベル₃と同じ。
 ・ FONT RESET 命令により、ひらがなの使用可。

MZ35 ・ 本文と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ レベル₃と同じ。

IBM55 ・ 漢字、ひらがな使用可。
 ・ 2 バイト・コードの文字がある。

PC100 ・ IBM55 と同じ。

MSX ・ レベル₃と同じ。

シンボル ? : ; , " ' ,

形式 : ? " タテ "

機能 : 特殊な意味をもった文字, 記号.

- 特徴 :
- 1 ? (疑問符) は PRINT 文の代わりに用いることができる.
 - 2 : (コロン) は命令文の区切りを示し, マルチ・ステートメント¹⁾に使用する.
 - 3 ; (セミコロン) は PRINT 文または LPRINT 文を改行せずに続けて出力する.
 - 4 , (コンマ) は PRINT 文または LPRINT 文中に用いると, 定められた領域に分けて出力する. また, 変数の区切りに用いる.
 - 5 " (ダブル・クォート) は (") 内が文字定数であることを表す.
 - 6 ' (アポストロフィ) は REM 文の代わりに用いることができる.

用語 : ¹⁾マルチ・ステートメント(マルチ文) 数種のステートメントを連続したもの. 多重命令ともいう.

プログラム例 : 1 行に複数の命令文を書く.

```
10 LPRINT "タテ"
20 LPRINT "ヨコ"
30 LPRINT "タテ":LPRINT "ヨコ"
40 END
```

実行例

```
タテ
ヨコ
タテ
ヨコ
```

参照 : PRINT, 文字定数

APL (') は REM 文として使用できない.

TRS_I (') は REM 文として使用できない.

CPM (') は REM 文として使用できない.

M223 (?) は PRINT 文の代用とはならない.
 (') で囲んでも文字列となる.
 マルチ・ステートメントは, 自動改行されて表示される.
 PI は円周率を表現.
 (') は REM 文として使用できない. REM 文の代用には / , ! が使える.

MZK (') は REM 文の代用にならない.
 PI は円周率 3.1415927 を表現.

CBM (') は REM 文の代用にならない.

PC8 (') は REM 文の代用にならない.

TRS_{II} (') は REM 文の代用にならない.

PC3 (:) はラベル名とステートメントの区切りにも使用可.
 (') で囲んでも文字列となる.
 PI は円周率を表現.
 (') は REM 文として使用できない. 代用には (!) が使える.

IF82 (') は REM 文の代用にならない.

レベル₃ (') は REM 文の代用にならない.

C180 (:) によるマルチ・ステートメントは使えない.
 (?) は PRINT 文の代用とはならない.
 (') は 2 進数値定数を示す.
 PI は円周率を表現.

(') は REM 文として使用できない. REM の代用には (!) が使える.

M243 (') は REM 文として使用できない.

MZB (') は REM 文として使用できない.

BUB (') は REM 文として使用できない.

FM8 (') は REM 文として使用できない.

PSP (') で囲んでも文字列となる.
 (?) は PRINT 文の代用とはならない. ただし, 短縮形 (P, PR, PRI, PRIN のいずれでもよい) で代用できる.

PC88 (') は REM 文として使用できない.

N52 (') は REM 文として使用できない.

PC6 (') は REM 文として使用できない.

MLT (') は REM 文として使用できない.

HC (') は REM 文として使用できない.

FP11 (?) は PRINT 文の代用にはならない.
 PI は円周率を表現.

SMC (') は REM 文として使用できない.
 (') は REM 文として使用できない. (アンダーライン) は CALL の代用.

MZ35 (') は REM 文として使用できない. ただし, 特徴 1 は不可.

X1Hu (') は REM 文として使用できない. なお, π または PAI (1) は円周率を表現する.

算術演算子 ^ - * / + - ¥

形式： $A \wedge 10$ $-A$ $10 * A$ $A / 10$ $A + 10$ $A - 10$

べき乗 否定(負号) 乗算 浮動小数
点の除算 加算 減算

$M \text{ ¥ } 10 \%$

整数除算 省略可

機能：与えられた値に対し、演算子に従った算術演算を行う。

- 特徴：**
- 1 演算の順序は、 \wedge 、 $-$ 、 $(*)$ 、 $/$ 、 $(+)$ 、 $-$ である。
 - 2 演算の順序を変える場合は、カッコを用いる。ただし、カッコ内では通常の演算順位である。
 - 3 演算子が連続する場合は、カッコで区切る。
 - 4 整数の除算は、円記号(¥)で示す。この場合オペランドは、整数に丸められ(−32768から32767の範囲内でなければならない)、商は整数に丸められる。
 - 5 整数除算の演算順位は、乗算と浮動小数点の除算の次である。

プログラム例：べき乗、負号、乗算、除算、加算、減算を計算し印字する。

```
10 A=2
20 LPRINT A^10;-A;10*A;A/10
   ;A+10;A-10
30 END
```

実行例

```
1024 -2 20 .2 12 -8
```

参照：MOD

APL ・ M223 と同じ。

TRS_I ・ べき乗は(↑)を用いる。
(↑)の代わりに([])が出る機械
があるが機能は同じ。
・ ほかは M223 と同じ。

CPM ・ 本文と同じ。ただし、整
数除算は(\)を用い、オペラン
ドは四捨五入される。

M223 ・ 本文と同じ。ただし、
(¥)による整数除算はない(整数
型の数値がない)。

MZK ・ M223 と同じ。ただし、
べき乗は(↑)を用いる。

CBM ・ MZK と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ CPM と同じ。

PC3 ・ M223 と同じ。

IF8₂ ・ 本文と同じ。ただし、整
数除算において扱われる数値は四
捨五入され、商は丸められる。

レベル₃ ・ 本文と同じ。

C180 ・ M223 と同じ。

M243 ・ M223 と同じ。

MZB ・ M223 と同じ。

BUB ・ 本文と同じ。

FM8 ・ IF8₂ と同じ。

PSP ・ べき乗は**でも可。
・ ¥による整数除算はなし。ただし、
整数除算は単精度となる。

PC88 ・ IF8₂ と同じ。

N52 ・ 整数除算において扱われ
る数値は、四捨五入されて整数に
なり、商は小数点以下が切り捨て
られる。

PC6 ・ M223 と同じ。

MLT ・ IF8₂ と同じ。

HC ・ 本文と同じ。

FP11 ・ 特徴5は異なり、整数除
算の順位は同等に扱われる。

SMC ・ IF8₂ と同じ。

MZ35 ・ 本文と同じ。ただし、¥
は使えない。

X1Hu ・ 本文と同じ。ただし、特
徴1は \wedge 、 $|*|/|$ 、 $-$ 、 $|+|$ 、 $-|$
の順。

MB16 ・ 本文と同じ。

IBM55 ・ N52 と同じ。

PC100 ・ 本文と同じ。

MSX ・ 特に指定がない場合、定
数は倍精度として扱われる。

関係演算子 < = >

形式: $A=B$ $A<>B$ $A><B$ $A<B$ $A>B$

等しい 等しくない 小さい 大きい

 $A<=B$ $A=<B$ $A>=B$ $A>=>B$

等しいか小さい 等しいか大きい

機能: 二つの値を比較するために用いる。

特徴: 1 比較の結果は真(-1)または偽(0)となる。
 2 一つの式の中で関係演算子¹⁾と算術演算子が同時に使われた場合、常に算術演算子が最初に実行される。
 3 左右の値は算術式でもよい。
 4 比較した後は、論理演算子で結合して論理式として用いることができる。

用語: ¹⁾演算子 計算記号、+、-などの算術演算子、<、>などの関係演算子、AND、ORなどの論理演算子がある。

プログラム例: A1とA2の値を比較する。

```
10 A1=10
20 A2=20
30 IF A1=A2 THEN GOTO 60
40 IF A1<A2 THEN GOTO 70
50 LPRINT"A1 > A2":GOTO 80
60 LPRINT"A1 = A2":GOTO 80
70 LPRINT"A1 < A2"
80 END
```

実行例

A1 < A2

参照: IF, LET, 文字列演算子, 論理演算子

APL ・ C180と同じ。

TRS_I ・ 本文と同じ。ただし、><は不可。

CPM ・ 本文と同じ。ただし、><、=<、=>は不可。

M223 ・ 本文と同じ。

MZK ・ C180と同じ。

CBM ・ CPMと同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 本文と同じ。ただし、真(1)、偽(0)のように結果の値が異なる。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。ただし、結果として何をもどすかは不明。

M243 ・ 本文と同じ。

MZB ・ C180と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ PC3と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。

SMC ・ 本文と同じ。

MZ35 ・ 本文と同じ。ただし、特徴1で真のときは(1)となる。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。

MSX ・ 本文と同じ。

文字列演算子 + = < >

形式： “タテ” + E\$ “タテ” < “ヨコ”

連結

比較

機能：文字列の連結や比較を行う。

- 特徴： 1 比較 (= < >) はそれぞれの文字列から、1文字ずつそのアスキー・コード¹⁾の比較を行う。もしすべてのアスキー・コードが等しければ、その二つの文字列は等しくなり、もしアスキー・コードが異なれば、小さいコードのものが数値的に小さいとされる。また、比較の途中で一方の文字列が終わった場合は、短い文字列の方が小さいとされる。
- 2 文字列の前後の空白も意味をもつ。
- 3 文字列をアルファベット順にならべたり、文字列の値を調べたりするのに用いると便利である。

用語： ¹⁾アスキー・コード (ASCII コード) アルファベット、数字、記号を8 (7ビット + パリティ1ビット) ビット・コードで表したコード。

²⁾連続コード 各システムで使用されるすべての文字に付けられている連続番号。

プログラム例： E1\$ と E2\$ を連結する。

```
10 E1$="タテ"
20 E2$="ヨコ"
30 E3$=E1$+E2$
40 LPRINT E3$
50 END
```

実行例

タテヨコ

参照：関係演算子

APL ・本文と同じ。

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・連結は不可。
・比較は同値 (=) だけ可。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・連結は不可。
・JIS コードで比較する。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・連結は (&) を用いる。
・比較の場合、その大小関係は、CRT 表示文字コード一覧で比較する。

M243 ・本文と同じ。

MZB ・MZK と同じ。ただし、比較では文字列の長さが優先される。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。ただし、連結された文字列を比較する場合、連結結果の先頭50文字が有効。

PC88 ・本文と同じ。

N52 ・JIS コード (ただし、漢字は内部コード) で比較する。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・本文と同じ。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。なお、ナル・ストリング (" ") は、どの文字列よりも小さい。

MB16 ・本文と同じ。

IBM55 ・比較は JIS コードおよび連続コード²⁾で行う。

PC100 ・N52 と同じ。

MSX ・本文と同じ。

論理演算子 NOT AND OR XOR IMP EQV

ノット, アンド, オア, エクスクルーシブ・オア, インプリケート, イクイバレント

形式: NOT A A AND B A OR B

否 定 論理積 論理和

A XOR B A IMP B A EQV B

排他的論理和 包 含 同 値

機能: ビット操作やブール演算を行ったり, いくつかの関係を調べたりする。

特徴: 1 一つの式の中では, 論理演算は算術・関係演算の後で行われる。
2 論理演算子はオペランドを16ビットの符号付き2の補数表示(*)の整数に変換してから演算を行う。指定された演算はこの整数に対し, ビットごとに対応して実行される。

A B	NOT A	A AND B	A OR B	A XOR B	A IMP B	A EQV B
1 1	0	1	1	0	1	1
1 0	0	0	1	1	0	0
0 1	1	0	1	1	1	0
0 0	1	0	0	0	1	1

- 3 IF文の, 条件を組み合わせるときなどに使われる。
4 XORは, A AND NOT B OR NOT A AND B, IMPは, NOT A OR B, EQVは, A AND B OR NOT A AND NOT Bで代用できる。

プログラム例: 10000₂ (10進数で16) と 11111₂ (10進数で63)の
論理積 (AND) を計算し印字する。

```
100002
AND 111112
100002 (10進数で16)
```

```
10 A1=16
20 A2=63
30 B=A1 AND A2
40 LPRINT B
50 END
```

実行例

16

参照: IF

APL ・ PC3と同じ。

TRS_I ・ AND, OR, NOTのみ使用可。

CPM ・ 本文と同じ。

M223 ・ @ AND @ または AND, @ OR @ または OR, @ XOR @, NOT @ のみ。

MZK ・ 論理演算子として AND は*, ORは+のみ。

CBM ・ TRS_Iと同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ AND, OR, NOTのみ。

IF8₂ ・ 本文と同じ。

レベル₃ ・ 本文と同じ。

C180 ・ AND% ('007F', '12C6'), EOR% ('007F', '1234'), NOT% ('0000'), OR% ('00FF', '1234') の形で使用可。
・ ビット処理のために, SBIT%, RBIT%, BIT, SL%, SR% 各関数が用意されている。

M243 ・ M223と同じ。

MZB ・ MZKと同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP —

PC88 ・ 本文と同じ。

N52 ・ NOT, AND, OR, XORのみ使用可。

PC6 —

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FPI1 ・ N52と同じ。

SMC ・ N52と同じ。

MZ35 ・ PC3と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。

MSX ・ 本文と同じ。

数値定数 (その1) 整数型定数

形式: 9% &H8F &O173
 └─┬─┘ └─┬─┘
 省略可 16進定数¹⁾ 省略可
 └─┬─┘ └─┬─┘
 10進定数 8進定数²⁾

機能: -32768から32767までの値。

特徴: 1 10進定数は0から9までの数字と+, -を用いて表現する。後に整数型であることを示す(%)をつけることができる。
 2 16進定数は0から9までの数字とAからFまでの英字, それと+, -を用いて表現する。前に16進定数を示す(&H)を付けなければならない。
 3 8進定数は0から7までの数字と+, -を用いて表現する。前に8進定数を示す(&) または(&O)を付けなければならない。
 4 符号を付けるときは一番先頭とする。

用語: ¹⁾16進数 16進法によって表された数値。16進法とは、0から9までの数字とAからFまでの英文字によってすべての数値を表す方法。10進数の27は16進数の1Aである。

²⁾8進数 8進数によって表される数値。8進法では、基数8を単位とし1桁上がある。☐ 2進数。

³⁾JIS16進定数 JIS16進コードを表す定数。

⁴⁾JIS区点定数 JIS区点コードを表す定数。

プログラム例: 10進定数10と16進定数&H10(10進数で16) 8進定数&O10(10進数で8)を印字する。

```
10 LPRINT 10; &H10; &O10
20 END
```

実行例

```
10 16 8
```

参照: 単精度型定数, 倍精度型定数

APL ・ 8進, 16進定数は不可。
 ・ 整数の範囲は-32767から32767。

TRS_I ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ 8進, 16進定数は不可。
 ・ 単精度, 高精度どちらでも整数型可。

MZK ・ 8進定数は不可。
 ・ 16進定数は(\$)を付けて表現。

CBM ・ 8進, 16進定数は不可。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 8進定数は不可。
 ・ 16進定数は(&)を付けて表現。
 ・ 整数の範囲は-32767から32767。

IF8₂ ・ 本文と同じ。

レベル₃ ・ 本文と同じ。

C180 ・ 8進定数は不可。
 ・ 10進定数は%をつけない。
 ・ (')で閉んだ4桁までの16進定数可。

M243 ・ M223と同じ。

MZB ・ MZKと同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 8進定数は不可。
 ・ 16進定数は(\$)を付けて表現する。
 ・ 整数範囲は-9999から9999。

PC88 ・ 本文と同じ。

N52 ・ 8進定数は不可。
 ・ 16進定数は(&)だけでも表現可。

PC6 ・ 8進定数は不可。

MLT ・ 8進定数は(&)のあとに0が必要。
 ・ 2進定数は(&)を付けて表現。

HC ・ 本文と同じ。

FP11 ・ 8進定数は(&)のあとに0が必要。

SMC ・ 8進定数は不可。

MZ35 ・ 8進は不可, 16進は&をつける。

X1Hu ・ 8進定数は(&)のあとに0が必要。
 ・ 2進定数は(&B)を付けて表現する。

MB16 ・ 本文と同じ。

IBM55 ・ ほかに, JIS16進定数¹⁾, JIS区点定数²⁾が使える。前者は(&J)後者は(&K)を付けて表現する。

PC100 ・ IBM55と同じ。

MSX ・ 2進定数は, 0か1の数字と, 前に(&B)を付けて表現する。

数値定数（その2） 単精度型定数

形式：3.14! 1.234E5

省略可 浮動小数点型式¹⁾

固定小数点型式²⁾

機能：有効桁数が7桁以下の数。

- 特徴：1 単精度型定数は、うしろに単精度を示す(!)を付けることができる。
- 2 単精度の場合は、7桁の精度で格納され四捨五入されて6桁まで表示される。すなわち有効桁数³⁾は6桁である。
- 3 固定小数点型式は、正または負の定数(0を含む)で、小数点を含む。
- 4 浮動小数点型式は、指数型式で表現された正または負の数値で、符号(なくても可)付きの整数または固定小数点型式の数値(仮数部)に続いて、文字"E"、そして符号(なくても可)付きの整数(指数部)で表現される。指数部の範囲は、-38から+38までである。

用語：¹⁾浮動小数点型式 演算を行うとき自動的に小数点を調節できる。数値を符号部、指数部、仮数部に分けて1語の中に入れて、演算を行う。実数型の数は浮動小数点型式である。

²⁾固定小数点型式 先頭の構成単位で符号を、そのつぎから数値そのものを表し、通常はその数値の前に小数点があるものとみなして演算を行う。整数型の数は通常、固定小数点型式である。

³⁾有効桁数(有効数字) 使用可能、あるいは演算可能な数値の範囲。

⁴⁾ダイレクト・モード このモードの場合、文および命令には行番号はつけず、命令が入力された時点で、すぐに実行される(コマンド・モードともいう)。□ プログラム・モード。

プログラム例：単精度型定数を入力し、印字する。

```
10 INPUT A
20 PRINT A
30 INPUT A
40 PRINT A
50 END
```

実行例

```
? 123456
123456
? 123456789
1.23457E+08
```

参照：倍精度型定数

APL ・有効桁数は9桁。
・指数部の範囲は-39から38。

TRS_I ・有効桁数は6桁。
・指数部の範囲は-39から38。

CPM ・本文と同じ。

M223 ・有効桁数は6桁。
・指数部の範囲は-78から75。

MZK ・有効桁数は8桁。
・指数部の範囲は-18から18。

CBM ・有効桁数は10桁。
・指数部の範囲は-39から38。

PC8 ・本文参照。

TRS_{II} ・有効桁数は7桁。
・指数部の範囲は-39から38。

PC3 ・有効桁数は12桁。
・指数部の範囲は-99から99。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・有効桁数15桁。
・指数部の範囲は-79から75。

M243 ・M223と同じ。

MZB ・有効桁数は16桁。
・指数部の範囲は-48から78。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・有効桁数は8桁。
・指数部の範囲は-62から62。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・有効桁数は9桁。
・指数部の範囲は-38から38。

MLT ・ダイレクト・モード⁴⁾時は自動的に表示方法、精度が切り替わる。

HC ・指数部の範囲は-38から35。

FP11 ・有効桁数は9桁。
・9桁の精度で格納され、6桁まで表示される。
・指数部の範囲は-99から99。

SMC ・有効桁数は14桁。
・指数部の範囲は-63から63。

MZ35 ・PC3と同じ。

X1Hu ・有効桁数は8桁。

MB16 ・TRS_{II}と同じ。

IBM55 ・6桁の精度であるが、7桁で格納され7桁で表示される。
・指数部の範囲は-39から38。

PC100 ・指数部の範囲は-62から62。

MSX ・有効桁数は6桁。
・指数部の範囲は-64から63。

数値定数 (その3) 倍精度型定数

形式: 3.14159265358979 # 1.234567890D12

省略可 浮動小数点型式

固定小数点型式

機能: 有効桁数が8桁以上の数。

- 特徴:
- 1 単精度型とまぎらわしいときは後に(#)をつけて区別する。
 - 2 倍精度の場合、17桁の精度で格納され、16桁まで表示される。
 - 3 固定小数点型式は単精度と表現は同じであるが、有効桁数が7桁以下のときは(#)をつけなければならない。
 - 4 浮動小数点型式は単精度と表現は同じであるが、指数部の(E)の代わりに(D)を使わなければならない。
 - 5 高い精度で計算するとき用いると便利。
 - 6 SINなどの関数は、入力の数として倍精度も使えるが、単精度化されて計算され、結果も単精度となる。

プログラム例: 倍精度型定数を入力し、印字する。

```
10 INPUT A#
20 PRINT A#
30 INPUT A#
40 PRINT A#
50 END
```

実行例

```
? 1234567890123456
1234567890123456
? 1234567890123456789
1.234567890123457D+18
```

参照: 単精度型定数

APL —

TRS_I ・有効数字16桁までの数。CPM ・IF8₂と同じ。

M223 ・必ず(#)を付けなければならない。
 ・指数部は(E)を使う。
 ・倍精度は16桁で高精度と呼ばれる。

MZK —

CBM ・指数部の範囲は-39から+38。
 ・10桁で格納され9桁まで表示。

PC8 ・本文参照。

TRS_{II} ・有効数字17桁までの数。

PC3 —

IF8₂ ・本文と同じ。ただし、16桁の精度で格納され16桁まで表示。

レベル₃ ・IF8₂と同じ。

C180 —

M243 ・M223と同じ。

MZB ・実数範囲±1E-48から±9.999999999999999E+78までのBCD型浮動小数点型式。

BUB ・本文と同じ。

FM8 ・本文と同じ。ただし、17桁の精度で格納され16桁まで表示。

PSP ・有効桁が9桁以上の数。
 ・14桁で格納され14桁まで表示。
 ・指数値は-62から62まで。

PC88 ・IF8₂と同じ。

N52 ・FM8と同じ。

PC6 —

MLT ・IF8₂と同じ。

HC ・PC88と同じ。

FP11 ・有効桁数が10から19桁の数。
 ・19桁の精度で格納され、16桁まで表示。
 ・指数部は(E)を使う。
 ・組み込み関数は倍精度まで計算可能、また結果も倍精度で表示。
 ・(&)を付記することにより、有効桁数が20〜29の倍々精度型定数として使用可。

SMC —

MZ35 —

X1Hu ・指数部の範囲は-38〜38。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・14桁精度14桁表示。
 ・指数表示では単精度と同じくEを用いる。

MSX ・有効数字14桁までの数。

文字定数

形式: " タテ "

機能: BASIC が実行時にそのまま使用する文字列。

特徴: 1 最大 255 文字までの引用符 (") に囲まれた英数文字の列。

プログラム例: 文字列 " タテ " と " ヨコ " を印字する。

```
10 LPRINT " タテ "; " ヨコ "
20 END
```

実行例

タテヨコ

参照: 変数

APL ・ 本文と同じ。

TRS_I ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ (') で囲んでもよい。
・ 最大長 249 文字。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ (') で囲んでもよい。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ @ で囲んだ16進数字でもよい。

M243 ・ M223 と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ (') で囲んでもよい。
・ 最大長は通常32。ただし、DIM 文で変更できる。

PC88 ・ 本文と同じ。

N52 ・ 最大 245 文字まで。

PC6 ・ 本文と同じ。

MLT ・ 文番号の使用状態で文字

数の上限は242~252まで変わる。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。

SMC ・ 本文と同じ。

MZ35 ・ PC3 と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ ほかにカナ文字、記号、漢字も用いることができる。

MSX ・ 本文と同じ。

FP11 ・変数名は最大 255 文字。
・先頭は英文字のほかにカナが使える。

変数

<p>英文字, カナ, 数字で構成される.</p> <ul style="list-style-type: none"> ・先頭から予約語の綴りを含んではいけない. ・属性文字 (&) は倍々精度変数を示す. 			
<p>SMC ・ PC88 と同じ. ただし, 変数名は最大 255 文字.</p> <ul style="list-style-type: none"> ・アンダーラインも使用可. 			
<p>MZ35 ・ 整数型は %, 実数型は #, 文字型は \$ で, @ でナルを代入できる.</p>			
<p>X1Hu ・ 変数名は最大240文字.</p> <ul style="list-style-type: none"> ・先頭から予約語のつづりを含んではいけない. 			
<p>MB16 ・ 変数は40字以内, 属性文字は変数の終わり以外不可.</p>			
<p>IBM55 ・ BUB と同じ.</p>			
<p>PC100 ・ PC88 に同じ.</p>			
<p>MSX ・ 本文と同じ. ただし, 属性文字が省略された場合は倍精度.</p>			

ラベル

形式: 50 * "EXIT" : PRINT "END"

文字定数

省略可

機能: 行番号の代わりに、名前(ラベル¹⁾)でその行を指定できる。

- 特徴: 1 GOTO 文の飛び先に文字定数や文字変数を使う。
 2 同じラベルを二つ以上の行につけてはならない。
 3 プログラムの下の方へ飛ばしたいとき、あらかじめ行番号はわからないが、ラベルを使うと便利である。また、フローチャート中につけた枝の名前と、ラベルの文字列を一致させると、デバッグ²⁾に便利である。
 4 予約語でもよい。

用語: ¹⁾ラベル(ラベル名) 行番号の代わりにつけられる、文字定数による行の名前。
²⁾デバッグ 正しい結果を得るまでに行うプログラムの修正作業のこと。

プログラム例: 2次方程式の根を実数解のときのみ求める。

```
10*"BEGIN":INPUT A,B,C
20 D=B*B-4*A*C
30 IF A=0 THEN GO TO "END"
40 PRINT A:"X^2+":B:"X+":C:"=0",
50 IF D<0 THEN GO TO "COMPLEX"
60 PRINT "REAL ":(-B-SQR(D))/(2*A)
: (-B+SQR(D))/(2*A)
70 GO TO "BEGIN"
80*"COMPLEX":PRINT "COMPLEX,INPUT AGAIN !"
90 GO TO "BEGIN"
100*"END":END
```

実行例

```
2 X^2+ 3 X+-9 =0 REAL -3 1.5
1 X^2+ 1 X+ 5 =0 COMPLEX,INPUT AGAIN !!
```

参照: GOTO, ON GOTO

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 ・本文参照。

IF₈₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 50 *EXIT:PRINT
"END"

- ・飛ぶときは、GOTO *EXIT。
- ・ラベル名は、英文字ではじめる。英数字、ビリオド(.)を続けることができる。大文字、小文字の区別はない。
- ・BASICの予約語は、ラベル名に使えない。中に含めばよい。
- ・ラベル名の長さは、1行が255文字以内におさまるならば、いくらで

- もよい。
- ・ラベル名の後の(:)は、スペースでもよい。
- ・LIST コマンドやDELETE コマンドの引数として使うこともできる。
- LIST *A — *B

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC ・PC88と同じ。

MZ35 ・本文と同じ。

X1Hu LABEL "EXIT"
 ・ラベル名は255文字までの文字列。
 ・同一ラベル名がある場合、最初に出てきたラベル名が優先される。

MB16 —

IBM55 —

PC100 ・予約語を用いてはならない。ただし、中に含んでもよい。

MSX —

配列

形式：B (100) B1 (10, 20)

添字 添字 * 省略・繰り返し可

添字

機能：同じ変数名で参照することのできる値の集まり、または表。

特徴: 1 添字をつけた配列は、変数と同じように扱える
2 配列変数名は、その配列の次元と同じ個数の添字をもつ。
3 添字の下限は 0。
4 添字の数を次元という。次元の大きさは 1 行の長さからの制限をうける。従って十分に大きな次元が可能。
5 DIM 宣言なしで用いたときは、添字は 10 まで。

プログラム例：配列 A (10) を宣言し、値を入力した後、印字する。

```
10 DIM A(10)
20 FOR I=1 TO 10
30 A(I)=I
40 NEXT I
50 FOR I=1 TO 10
60 LPRINT A(I);
70 NEXT I
80 END
```

实行例

1 2 3 4 5 6 7 8 9 10

参照：DIM

APL ・次元は88次元まで.

TRS₁ ・本文と同じ.

CPM ・本文と同じ.

M223 ・数値配列の次元は2次元まで、添字は行、列ともメモリ容量の最大まで、添字の下限は0。
・文字配列の次元は1次元まで、行の添字はメモリ容量の最大まで、列の添字は255まで。

MZK ・次元は2次元まで、添字は0から255まで。
・特徴5は不可、

CBM ・次元は 255 次元まで、添字の上限は 32767.

PC8 · 本文参照.

TRS_{II} ・ PC3 と同じ.

PC3 ・特徴5は不可.

IF82 ・次元は 255 次元まで、添字の上限は 32767.

レベル₃ ・本文と同じ.

C180 ・次元は2次元まで。
・2進数値配列も許される。

M243 ・ M223 と同じ.

MZB ・ MZK と同じ.

BUB ・本文と同じ.

FM8 ・本文と同じ。ただし、添字の上限は、整数型式で 32766。

PSP ・次元は15次元まで.

PC88 ・次元は 255 次元まで.

N52 ・次元は3次元まで。
・添字の上限は32767.

PC6 ・次元は3次元まで。
・添字は1から2447まで。

MLT ・ CBM と同じ。

HC ・本文と同じ。

FP11 ・添字の最大値は記憶容量の許す限り、
・次元は表記上許される限りの多次元まで。

SMC ・次元は10次元まで.

MZ35 ・本文と同じ。ただし、特徴3の下限は1、特徴5は不可。

X1Hu ・次元は255次元まで、添字は最大255個、上限は32767.

MB16 ・本文と同じ。

IBM55 ・ CBM と同じ。

PC100 ・ PC88 と同じ。

MSX ・ PC88 と同じ.

型变换

形式： $M\% = 12.34$ $A = 12\%$ $C\# = A$
 └──────────┘ └──┘ (倍精度型への変換)
 12が格納(整数型への変換) (単精度型への変換)

機能：必要に応じて、数値定数の型をある型からほかの型に自動的に変換する。

- 特徴: 1 ある型の数値変数が違った型の数値変数に割り当てられた場合、その変数名によって宣言された型に変換して格納される。
- 2 式¹⁾を計算する場合は、算術または論理演算²⁾のオペランド³⁾はすべて同じ精度(最も精度の高いオペランド)にそろえられる。同様に、算術演算⁴⁾の結果もこの精度で得られる。
- 3 論理演算では、オペランドは整数に変換され整数の結果が得られる。
- 4 浮動小数点型式で表された数値が整数に変換される場合には、小数部分は切り捨てられる。
- 5 倍精度が単精度に変換される場合は、最初の7桁目を丸めたものが有効。
- 6 型変換の組み合わせは次のとおり。

变换前 \ 变换后	整 数 型	单精度型	倍精度型
整 数 型		(1)	(2)
单 精 度 型	(3)		(4)
倍 精 度 型	(5)	(6)	

(1)と(2)の場合には、 $0.999 \rightarrow 1$ 、 $-0.01 \rightarrow 0$ のように整数化される。(6)の場合は、 $0.999 \rightarrow 0.99899995326995585$ のように誤差が入るので注意。

- 用語: ¹⁾式 変数に文字、または数値の定数、または変数、あるいはある値を得るために定数や変数を演算子で結ぶこと。
²⁾論理演算 論理演算子 (NOT, AND, OR など) を使って、その演算子の論理に従って1ビットごとに演算すること。
³⁾オペランド 命令の対象となる数値、命令においてその対象となることを示す部分、あるいは、対象となる数値のはいっているアドレスを示す言葉、ここでは計算に用いる数値のことである。
⁴⁾算術演算 算術演算子 (+, -, *, / など) を使って四則演算を行うこと。

プログラム例：浮動小数点型式で示された数値12.56を整数に変換し、印字する。

```
10 A%=12.56
20 LPRINT A%
30 END
```

実 行 例
12

参照：整数型定数，单精度型定数，倍精度型定数

型変換

<p>APL ・整数に変換する場合、小さいほうへ丸められる結果、$A\% = -0.01$では-1となるので注意。</p>	<p>MZB —</p>		
<p>TRS_I ・式中の一つの定数が倍精度のとき倍精度となる。 ・数字が倍精度でなく、-32768から32767の範囲外ならば単精度となる。 ・上記のいずれでもない場合は整数となる。</p>	<p>BUB ・本文と同じ。ただし、詳細は不明。</p>		
<p>FM8 ・IF8₂と同じ。</p>	<p>PSP —</p>		
<p>CPM ・本文と同じ。</p>	<p>PC88 ・本文と同じ。ただし、実数から整数へ変換されるとき、小数点以下は四捨五入される。</p>		
<p>M223 ・単精度数を高精度数へ変換はDCML (X)。 ・高精度数は単精度数値内での変換可FLOT (X#)。</p>	<p>N52 ・整数の除算と剰余および論理演算では、オペランドは整数に変換され整数の結果が得られる。</p>		
<p>MZK —</p>	<p>PC6 —</p>		
<p>CBM ・本文と同じ。</p>	<p>MLT ・PC88と同じ。</p>		
<p>PC8 ・本文参照。</p>	<p>HC ・本文と同じ。</p>		
<p>TRS_{II} ・数値間ではいかなる変換も可。</p>	<p>FP11 ・本文と同じ。</p>		
<p>PC3 ・数値関数MODIFY, CDMS, CDEGによって行う。</p>	<p>SMC ・PC88と同じ。ただし、倍精度型はない。</p>		
<p>IF8₂ ・本文と同じ。ただし、浮動小数点型式で表された数値が整数に変換される場合、小数点以下は四捨五入される。</p>	<p>MZ35 ・整数と実数については可。</p>		
<p>ベル₃ ・単精度から倍精度に変換される場合は、最初の7桁目を丸めたものが有効。</p>	<p>X1Hu ・本文と同じ。</p>		
<p>C180 ・数値関数FIX %とFLTによって行う。</p>	<p>MB16 ・PC88と同じ。</p>		
<p>M243 ・M223と同じ。</p>	<p>IBM55 ・PC88と同じ。</p>		
	<p>PC100 ・PC88と同じ。</p>		
	<p>MSX ・本文と同じ。</p>		

スクリーン・エディタの使い方

機能：カーソル移動キーおよび編集キーを使うことにより、画面上の文字を自由に編集することができる。

特徴：スクリーン・エディタ¹⁾により入力・編集した画面上の内容は **RETURN** キーを入力しなければ有効とはならない。

使い方：1 カーソルの移動

(a) **↑**, **↓**, **←**, **→** これら四つのキーは、それぞれの方向にカーソルを1字分移動させる。

(b) **CTL-N** このキーは、カーソルを次の項の位置まで移動させる。

(c) **CTL-B** このキーは、カーソルを前の項の先頭まで移動させる。

2 挿入と削除

(a) **INS** または **CTL-R** このキーは、カーソル位置の左側に空白を挿入することができる(カーソルの位置は動かない)。

(b) **DEL** または **CTL-H** このキーは、カーソル位置を左側へ後退させ1字まっ消する。

3 行の分割と結合

(a) **CTL-J** このキーをカーソルが分割しようとする位置で入力することにより、そのプログラム行のカーソル位置から後が画面の次の行の最初に移動する。

(b)画面上で連続した2行を1行のプログラム文にするには、まずカーソルを始めの行の1番最後に移動し、そこが空白なら空白を、文字があるならその文字を入力する。これで元の2行は論理的には1行となる。

用語：¹⁾スクリーン・エディタ 画面を用いた編集、従来のタイプライタ方式のライン・エディタによりプログラムの作成および修正が容易である。

²⁾オート・リピート キーを押し続けると、何度も押さなくても自動的に同一文字が入力される機能。

実行例：プログラム中の10番の綴りを修正する。

```
list
10 ORINT A
20 PRINT B
30 END
Ok
list 10-10
10 PRINT A
list
10 PRINT A
20 PRINT B
30 END
Ok
```

参照：コントロール文字

スクリーン・エディタの使い方

<p>APL ・カーソルの移動は、上は (ESC)-[I]、下は (ESC)-[M]、右は (ESC)-[K]、左は (ESC)-[J]。エディットの始めは左はじの行番号の所へカーソルを移動し修正し、文字のない所までカーソルを移動し (CR) を入力する。</p>	<p>・文字挿入は (IV) キー、文字削除は (V) キーを用いる。 ・(NL) キーを入力しなければ入力・編集した内容は有効とはならない。</p>	<p>をその行の先頭に移動させる。</p>	<p>削除は本文 (DEL) キーの代わりに (BACK SPACE)、(DEL) キーはカーソル位置を消去しカーソルは移動しない。</p>
<p>TRS_I —</p>	<p>M243 ・M223 と同じ。</p>	<p>PC6 ・1 (b) は不可、および3 (a) はライン・フィード。</p>	<p>IBM55 ・カーソル移動、次の項、前の項への移動は MB16 と同じ。 ・文字の削除には (削除)、(後退) キー。文字の挿入は (挿入) キー。タブ位置移動は (TAB) キー、カーソルのある行の消去は (ESC)。</p>
<p>CPM —</p>	<p>MZB ・MZK と同じ。ただし、カーソルは [↑]、[↓]、[←]、[→] キーを用い、オート・リビート付き (SHIFT) を押し、その後各カーソル・キーを押す。</p>	<p>MLT ・[]：バック・スペース。 ・オート・リビート付。 ・(ERASE LINE)：1 行消去。 ・1 (b)、2 (b)、3 (a) 以外本文と同じ。</p>	<p>PC100 ・(CTL) + [F] で次の項、(INS) に関しては FM8 と同じ。ただし、カーソル・キーを使うと INS から抜ける。</p>
<p>M223 ・1 ステートメントのみスクリーン・エディット可。</p>	<p>BUB ・カーソルの移動は [↑]、[↓]、[←]、[→] キーを用いる。 ・文字挿入は (INS) キー、文字削除は (DEL) キーを用いる。</p>	<p>HC ・カーソル移動は [↑]、[↓]、(SHIFT) + [↑]、(SHIFT) + [↓] の四つで行う。 ・編集については、コントロール文字参照。</p>	<p>MSX ・(INS)、(HOME)、(DEL)、(BS) キー、カーソル移動キーにより修正。</p>
<p>MZK ・カーソルの移動は、カーソル・キーを用いる。 ・(CR) キーを入力しなければ入力・編集した内容は有効とはならない。</p>	<p>FM8 ・カーソルの移動はコントロール文字の項参照。 ・(INS) キーを1度押すと次に (INS) キーを押すまで文字の挿入可能。</p>	<p>FP11 ・(DEL) キーではカーソル位置は動かない。</p>	<p>SMC ・(RUBOUT)、(BS)、[←]、[↑]、[↓]、(HOME)、(CLS)、(DEL)、(INS)、(SHIFT)、(CTL) により修正する。</p>
<p>CBM ・カーソルの移動は (CRSR→)、(CRSR←)、(CRSR↑)、(CRSR↓)、キーを用いる。 ・文字挿入は (INST) キーを用いる。</p>	<p>PSP ・入力した行の修正は [→]、[←]、(DEL)、(INS)、(CLS) キーにより可能。</p>	<p>SI11 ・(DEL) キーではカーソル位置は動かない。</p>	<p>MB16 ・カーソル移動 (CTL) + [F] または (SHIFT) + [→] で次の項、前の項は本文と (SHIFT) + [←]、</p>
<p>PC8 ・本文参照。 ・長い距離を移動する際はキーボードのオート・リビート²⁾機能(押し続けると働く)を有効に活用できる。</p>	<p>PSC88 ・(INS) キーは FM8 と同じ。</p>	<p>MZ35 ・カーソルの移動はカーソル・キーを用いる。 ・文字挿入は (INS) キー、文字削除は (DEL) キーを用いる。</p>	<p>X1Hu ・次項移動 (CTL) + [F]、行の結合 (CTL) + [W]、タブ設定 (CTL) + [T]。 ・挿入は (CTL) + [A] で INS モードになり、次に (CTL) + [A] が入力されるまで実行される。 ・オート・リビートは REPEAT OFF で解除、ON で復帰。 ・CLICK ON でクリック音を出す。OFF は止める。</p>
<p>TRS_{II} —</p>	<p>N52 ・カーソルを次の項・前の項の先頭まで移動させるには、(CTL)-[→]、(CTL)-[←] を使う。 ・[↑] キーはカーソルを次のプログラム行の先頭に移動させ、もしカーソルがプログラム行の先頭にあるときに押された時は一つ前のプログラム行の先頭に移動させる。 ・[↓] キーはカーソルを次のプログラム行の先頭に移動させる。 ・[K] キーはカーソルを画面の左上隅に移動させる。 ・(_上DEL) キーはカーソルがある行のすべての文字を消去し、カーソル</p>	<p>MB16 ・カーソル移動 (CTL) + [F] または (SHIFT) + [→] で次の項、前の項は本文と (SHIFT) + [←]、</p>	<p>削除は本文 (DEL) キーの代わりに (BACK SPACE)、(DEL) キーはカーソル位置を消去しカーソルは移動しない。</p>
<p>PC3 ・今入力した行と直前の行は (CL)、[↑]、[←]、[→]、(INS)、(DEL) キーで編集可。</p>	<p>IF82 ・本文と同じ。</p>	<p>MB16 ・カーソル移動 (CTL) + [F] または (SHIFT) + [→] で次の項、前の項は本文と (SHIFT) + [←]、</p>	<p>削除は本文 (DEL) キーの代わりに (BACK SPACE)、(DEL) キーはカーソル位置を消去しカーソルは移動しない。</p>
<p>レベル₃ ・本文と同じ。</p>	<p>C180 ・カーソルの移動は [↑]、[↓]、[←]、[→] キーを用いる。</p>	<p>MB16 ・カーソル移動 (CTL) + [F] または (SHIFT) + [→] で次の項、前の項は本文と (SHIFT) + [←]、</p>	<p>削除は本文 (DEL) キーの代わりに (BACK SPACE)、(DEL) キーはカーソル位置を消去しカーソルは移動しない。</p>

コントロール文字

機能：コントロール・キーと同時に押して用いる文字。

特徴：1 各キーの意味は次のとおり。

- (a) **CTL**¹⁾-**B** カーソル²⁾を後退させる。
 (b) **CTL**-**C** プログラムの実行を中断する(**STOP** キーと同じ)。
 (c) **CTL**-**E** カーソル位置から以後を消去。
 (d) **CTL**-**G** ベルを鳴らす。
 (e) **CTL**-**H** 1文字消去。
 (f) **CTL**-**I** カーソルを次の8文字境界まで消す。
 (g) **CTL**¹⁾-**J** 行を二つに分ける。
 (h) **CTL**-**K** カーソルをホーム位置へ移す。
 (i) **CTL**-**L** 画面の消去。
 (j) **CTL**-**M** キャリッジ・リターン³⁾。
 (k) **CTL**-**N** カーソルを前進させる。
 (l) **CTL**-**R** 1文字挿入。

用語：¹⁾ **CTL (CTRL)** CONTROL の略。**CTL** キーはカーソル移動などの特殊な機能であるコントロール・キャラクタを入力するために用意されている。ほかの文字と同時に押す。

²⁾ **カーソル** CRT 上で次に入力される位置を示し、点滅表示の機種が多い。
³⁾ **キャリッジ・リターン** BASIC においてはコンピュータ本体に、現在 CRT 上のカーソルがある位置の行を入力すること。

⁴⁾ **デュアル・ルーチン** まったく同じルーチンを二つ用意しておき、それぞれに同じ仕事をさせるルーチンのこと。

⁵⁾ **ライン・フィード** 行を次の行に改行すること。

⁶⁾ **仮想スクリーン** コンピュータ内に大きな画面があると考え、LCDディスプレイ上で見えている画面は、その大きな画面の一部分を見ていることになる。この大きな画面を仮想スクリーンという。

⁷⁾ **バック・スペース** カーソルの左側を空白(削除)にして空白分だけ左側にもどること。

⁸⁾ **タブ** あらかじめ設定された位置をタブ(TAB)という。例えば、8桁ごとのカーソルの位置。

プログラム例：—

APL ・実行中断(**CTL**)-**C**、1
行削除(**CTL**)-**X**。

TRS： —

CPM ・エディット・モード(**CTL**)-**A**。

- ・実行、中断(**CTL**)-**C**、停止(**CTL**)-**S**、再開(**CTL**)-**Q**、出力停止(**CTL**)-**O**。
- ・後退まっ消(**CTL**)-**H**、境界まで消去(**CTL**)-**I**、入力行消去(**CTL**)-**U**、入力行再表示(**CTL**)-**R**、ベル(**CTL**)-**G**。

M223 ・画面、リバース(**CTL**)-**V**、ノーマル(**CTL**)-**U**、消去(**CTL**)-**L**。
 ・次の文番号の編集に移る(**CTL**)-**F**、(**CTL**)-**K**はその文をコピーする。

MZK —

CBM —

PC8 ・本文参照。

TRS¹⁾ ・カーソル、点滅(**CTL**)-**A**、消去(**CTL**)-**B**、点滅しない(**CTL**)-**D**、後退まっ消(**CTL**)-**H**、ホーム位置(**CTL**)-**T**、前行に行にもどす(**CTL**)-**K**。
 ・境界まで消去(**CTL**)-**I**、行消去(**CTL**)-**W**。
 ・画面、消去(**CTL**)-**X**、白文字(**CTL**)-**Y**、黒めき(**CTL**)-**Z**、行送り(**CTL**)-**J**、復帰(**CTL**)-**M**、デュアル・ルーチン⁴⁾、ON(**CTL**)-**N**、OFF(**CTL**)-**O**。

PC3 ・最も低いプログラム・ラインからの実行(**CTL**)-**DEB**。

IF8： ・カーソル、右(**CTL**)-**F**、

左(**CTL**)-**B**、ホーム位置(**CTL**)-**K**。

- ・後退まっ消(**CTL**)-**H**、以後文字消去(**CTL**)-**E**、1文字挿入(**CTL**)-**R**、境界まで消去(**CTL**)-**I**、行二分(**CTL**)-**J**。
- ・画面、消去(**CTL**)-**L**、出力の有効・無効(**CTL**)-**O**、復帰(**CTL**)-**M**。
- ・実行中断(**CTL**)-**C**、ベル(**CTL**)-**G**。

レベル³⁾ ・カーソル、ホーム位置(**CTL**)-**K**、後退まっ消(**CTL**)-**H**、境界まで消去(**CTL**)-**I**、次の項の先頭(**CTL**)-**F**、前の項の後尾(**CTL**)-**B**、1文字挿入(**CTL**)-**R**、画面消去(**CTL**)-**L**。
 ・タブ位置、セット(**CTL**)-**T**、解除(**CTL**)-**Y**。
 ・以後文字消去、行(**CTL**)-**E**、最下段(**CTL**)-**Z**。

C180 ・継続行表示(**CTL**)-**C**。
 ・行消去(**CTL**)-**A**、画面消去(**CTL**)-**L**、1文字消去(**CTL**)-**H**。
 ・実行停止(**CTL**)-**W**、実行中断(**CTL**)-**N**、New Line (**CTL**)-**Z**、ベル(**CTL**)-**G**。

M243 ・M223と同じ。

MZB —

BUB ・カーソル、右(**CTL**)-**F**、左(**CTL**)-**B**、下(**CTL**)-**J**、ホーム位置(**CTL**)-**K**。
 ・後退まっ消(**CTL**)-**H**、境界まで消去(**CTL**)-**I**、1文字挿入(**CTL**)-**R**、以後文字消去(**CTL**)-**E**、ベル(**CTL**)-**G**。
 ・実行、中断(**CTL**)-**C**、停止(**CTL**)-**S**。
 ・画面消去(**CTL**)-**L**、復帰

参照：スクリーン・エディタの使い方

コントロール文字

<p>(CTL) - [M].</p> <p>FM8 ・カーソル, 右 (CTL) - [Y], 左 (CTL) - [J], 上 (CTL) - [↑], 下 (CTL) - [↓].</p> <p>・後退まっ消 (CTL) - [H], 以後文字消去 (CTL) - [E], 境界まで消去 (CTL) - [I].</p> <p>・項への移動, 右から (CTL) - [B], 左から (CTL) - [F], 上から (CTL) - [Z], 下から (CTL) - [V].</p> <p>PSP ・画面コピー (CTL) - [COPY].</p> <p>・実行停止 (CTL) - [STOP].</p> <p>PC88 ・ヘルプ・キー (CTL) - [A], 1語削除 (CTL) - [D], 前の語へ進む (CTL) - [F].</p> <p>・表示, 無効 (CTL) - [O], 1時停止 (CTL) - [S].</p> <p>・行削除 (CTL) - [U], 行の後尾へ (CTL) - [X].</p> <p>・ほかはPC8と同じ, ただし, (CTL) - [N]はできない.</p>	<p>・左側文字1字の消去(バック・スペースと同じ) (CTL) - [H].</p> <p>・カーソルを次の8文字境界まで消す(TABキーと同じ) (CTL) - [I].</p> <p>・カーソル, 右 (CTL) - [Y], 左 (CTL) - [J], 上 (CTL) - [↑], 下 (CTL) - [↓] と (CTL) - [←], ホーム位置 (CTL) - [K], 行の右端 (CTL) - [N].</p> <p>・1行消去 (CTL) - [V] と (CTL) - [I].</p> <p>・画面消去 (CTL) - [L].</p> <p>・キャリッジ・リターン (CTL) - [M].</p> <p>・1文字挿入 (CTL) - [R].</p>	<p>(CTL) - [P], 共に40文字モードのみ動作.</p> <p>・1文字消去 (CTL) - [Q], 後退消去 (CTL) - [H], 行消去 (CTL) - [X].</p> <p>・エンター (CTL) - [W].</p> <p>・ほかは本文と同じ, ただし, (CTL) - [C], (CTL) - [J]は不可.</p> <p>SMC ・カーソル, 下 (CTL) - [J], ホーム位置 (CTL) - [T], タブ位置 (CTL) - [I].</p> <p>・1文字消去 (CTL) - [H].</p> <p>・キャリッジ・リターン (CTL) - [M].</p> <p>・一時停止 (CTL) - [S].</p> <p>・1行あける (CTL) - [O].</p> <p>・カーソル以後の行を1行下げる (CTL) - [Q].</p> <p>・これ以後に表示される文字がブリント出力する (CTL) - [P].</p>	<p>(CTL) - [U], 行消去 (CTL) - [E], 最下段まで消去 (CTL) - [Z], 最後に入力したコマンド表示 (CTL) - [A].</p> <p>IBM55 ・長い論理行入力 (CTL) - [改行], 1ワード右 (CTL) - [→], 1ワード左 (CTL) - [←], 以後文字消去は行が (CTL) - [S], 以降の画面消去が (CTL) - [↓], システム・リセットが (CTL) - [前面キー - 削除].</p> <p>PC100 ・PC88に同じ, ただし, (CTL) - [O]の機能はなく, ほか (CTL) - [T]はKEY ON, (CTL) - [W]は1ワード削除, (CTL) - [X]はロール・アップ, (CTL) - [Y]はロール・ダウン, (CTL) - [Z]はカーソル位置以降の画面表示の消去.</p> <p>MSX ・カーソル以後を消去 (CTL) - [E], 音出し (CTL) - [G].</p> <p>・その他は機種により異なる.</p>
<p>N52 —</p> <p>PC6 ・カーソルを1項目ごと右へずらす (CTL) - [F].</p> <p>・キャリッジ・リターン (CTL) - [M].</p> <p>・ライン・フィード⁵⁾ (CTL) - [J].</p> <p>・カーソルのある行を1行まっ消 (CTL) - [V].</p> <p>・1文字挿入 (CTL) - [R].</p> <p>MLT ・前の語の先頭へ (CTL) - [B].</p> <p>・プログラムの中断(BREAKと同じ) (CTL) - [C].</p> <p>・カーソルより右側を消去 (CTL) - [E].</p> <p>・次の行の先頭へ (CTL) - [F].</p> <p>・ブザー (CTL) - [G].</p>	<p>HC ・実画面, 仮想スクリーン⁶⁾の左端へ (CTL) - [A], 仮想スクリーンの右端へ (CTL) - [F], 上 (CTL) - [P], 下 (CTL) - [Q], 左 (CTL) - [S].</p> <p>・カーソル, 右 (CTL) - [D], ON (CTL) - [V], OFF (CTL) - [W].</p> <p>・AUTOモードから抜け出す (CTL) - [C].</p> <p>・カーソル右側消去 (CTL) - [E].</p> <p>・カーソル右側以後最下行までの消去 (CTL) - [Z].</p> <p>・バック・スペース⁷⁾ (CTL) - [H].</p> <p>・8文字ごとタブ⁸⁾ (CTL) - [I].</p> <p>・ライン・フィード (CTL) - [J].</p> <p>・カーソル, ホーム位置 (CTL) - [K].</p> <p>・仮想スクリーンのクリア (CTL) - [L].</p> <p>・キャリッジ・リターン (CTL) - [M].</p> <p>・挿入 (CTL) - [R].</p>	<p>MZ35 ・ (CTL) - [1] でテキスト画面のハード・コピー.</p> <p>X1Hu ・INSモード設定, 解除 (CTL) - [↑], 1ワード右 (CTL) - [F], タブ (CTL) - [I], 挿入 (CTL) - [R], 一時停止 (CTL) - [S], タブ設定 (CTL) - [T], 解除 (CTL) - [Y], カーソル以下画面クリア (CTL) - [Z], カーソル右 (CTL) - [←], 左 (CTL) - [J], 上 (CTL) - [↑], 下 (CTL) - [↓], (CTL) - [O] ~ [7] 背景色変更, テンキーで (CTL) - [O] ~ [7] 文字色変更.</p>	<p>MB16 ・カーソル, ホーム位置 (CTL) - [K], 後退まっ消 (CTL) - [H], 直後の1文字消去 (CTL) - [W], 画面消去 (CTL) - [L], 前の項の先頭 (CTL) - [B], 次の項 (CTL) - [F], 次の水平タブ (CTL) - [I], 行末 (CTL) - [N], 1行消去</p>

プログラムの実行 RUN

ラン

形式: ① RUN 70

② RUN "1:AFILE", R

省略可

機能: ① メモリの中にあるプログラムを実行する。
 ② ディスクから、ファイルをメモリにロードし実行する。

特徴: 1 ①で行番号を指定すると、その行から実行し、指定しないと最初から実行する。
 2 プログラムの実行を終了すると、コマンド・レベルにもどる。
 3 ①では実行前に開いているファイルはすべて閉じる。
 4 ②の指定順は、ドライブ¹⁾・ナンバ、(:), ファイル名、(,) R オプション²⁾である。
 5 ドライブ・ナンバを省略すると、1を指定したことになる。
 6 Rオプションを省略したときは、ロードする前にメモリの内容をすべてクリアし、開いているファイルを閉じる。Rオプションを指定したときは、開いているファイルはそのまま、すべてのデータ・ファイルは開いたままである。

用語: ¹⁾ドライブ フロッピー・ディスクヘデータを書いたり、データを取り出したりする装置。複数のドライブがある場合は、ドライブ番号をつけ、識別する。
²⁾オプション 目的にそって機能を選択するためにつける記号。

実行例: ファイル名"EXAMPLE"のプログラムをメモリにロードし実行する。次にメモリのプログラムを実行する。

```
run "2:EXAMPLE"
A=? 7
7^2 = 49
A=? -16
-16^2 = 256
A=? 0
Ok
LIST
10 INPUT "A=";A
20 IF A = 0 THEN GOTO 50
30 PRINT A;"^2=";A^2
40 GOTO 10
50 END
Ok
run
A=? 0
Ok
```

参照: LOAD, SAVE, CONT

APL RUN "AFILE", S7, D1, V1

②指定順は、ファイル名、スロット・ナンバ、ドライブ・ナンバ、ボリューム・ナンバ、

TRS: ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。ただし、
 ②RUN "AFILE" 70でAFILEの70行から実行できる。

MZK ・本文の①と同じ。

CBM ・MZKと同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 RUN n or s

・ライン・ナンバ n, またはライン・レベル s から実行。RUN の後に (/) を指定すると、実行モードに入る。このとき **DEB** キーによってデバッグできる。

IF8₂ ・本文と同じ。レベル₃ ・MZKと同じ。

C180 ② RUN AFILE . IB / 1
 ・ファイル名を指定すると、指定されたIBファイルを実行、すべて省略するとコード・ワーク中のプログラムを実行、コード・ワーク中にプログラムのないときは、ソース・ワーク中のプログラムをIBコードになおして実行。

M243 ・M223と同じ。

MZB ② RUN FD1@1,

"AFILE"

・指定順は、ドライブ・ナンバ、スレーブ・ディスク・ボリューム・ナンバ、ファイル名。

BUB ・本文と同じ。

FM8 ・本文と同じ。ただし、Rオプションはない。バブル (0) の場合はドライブ・ナンバの代わりに BUB0 : を指定する。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・ドライブ・ナンバを省略すると 0 を指定したことになる。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・ドライブ・ナンバは A, B, C, D のいずれかにする。
 ・RUN "COM0: (BLPSC)" とすると、RS-232C ポートの接続条件 (BLPSC) を設定し、プログラムのロード後実行する。
 ・プログラムはアスキー形式のものに限る。

FPI1 ・N52と同じ。

SMC ・②の最初の指定は装置名で省略すると現在使用中のディスクよりロードし実行する。

MZ35 ・本文と同じ。ただし、機能②は使えない。

X1Hu ・本文と同じ。ただし、②のオプションはない。

MB16 ・本文と同じ。

プログラムの実行 RUN

ラン

<p>IBM55 ・本文と同じ。ただし、ドライブ・ナンバは、A、B、Cのほか各装置を指定できる。</p> <p>PC100 ・ファイル・スベックは拡張子に BAS が必要。ドライブ名、パス名が省略されたときは、カレント・ディレクトリが指定される。</p> <p>MSX ・規格は①のみであるが、②の形式で使える機種がある。</p>			
--	--	--	--

トレース文 TRON, TROFF

トレース・オン, トレース・オフ

<p>形式：① TRON ② TROFF</p> <p>機能：①プログラムが実行しようとする行番号を順次出力(トレース¹⁾)する。 ②そのモードを解除する。</p> <p>特徴：1 TRON 文の実行を解除するには、TROFF 文、または、NEW コマンドを実行するとよい。 2 プログラムのデバッグ時に使うと便利である。</p> <p>用語：¹⁾トレース プログラムをデバッグするために行うことで、1ラインごとのプログラムの流れと、その実行結果が逐次表示される。</p>	<p>APL TRACE, NOTRACE</p> <p>TRS₁ ・本文と同じ。</p> <p>CPM ・本文と同じ。</p> <p>M223 —</p> <p>MZK —</p> <p>CBM —</p> <p>PC8 ・本文参照。</p> <p>TRS_{II} ・本文と同じ。</p>	<p>MLT ・本文と同じ。</p> <p>HC ・本文と同じ。</p> <p>FP11 ・LTRON でプリンタにトレースし、TROFF で解除する。</p> <p>SMC ・DEBUG コマンドで同じことができる。</p> <p>MZ35 ・PC3 と同じ。</p> <p>X1Hu ・本文と同じ。</p> <p>MB16 ・本文と同じ。</p>
<p>プログラム例：トレースを実行し、I=2 のならトレースを解除し、I=4 になったらトレースを実行する。</p> <pre> 10 FOR I=1 TO 5 20 PRINT I 30 IF I=2 THEN TROFF 40 IF I=4 THEN TRON 50 NEXT I 60 END </pre> <p>実行例</p> <pre> tron Ok run [10][20] 1 [30][40][50][20] 2 [30] 3 4 [50][20] 5 [30][40][50][60] Ok </pre> <p>参照：STEP ON, STEP OFF</p>	<p>PC3 TR ON M1, M2 ・M1 行から M2 行までトレースを行う。省略した場合は、すべての行について行う。TR OFF で解除。 ・TR PRINT M1 M2 でプリンタに印字する。</p> <p>IF8₂ ・本文と同じ。</p> <p>レベル₃ ・本文と同じ。</p> <p>C180 —</p> <p>M243 —</p> <p>MZB —</p> <p>BUB ・本文と同じ。</p> <p>FM8 ・本文と同じ。</p> <p>PSP ・本文と同じ。</p> <p>PC88 ・本文と同じ。</p> <p>N52 ・本文と同じ。</p> <p>PC6 —</p>	<p>IBM55 ・本文と同じ。</p> <p>PC100 ・本文と同じ。</p> <p>MSX ・本文と同じ。</p>

ステップ実行 STEP ON, STEP OFF

ステップ・オン, ステップ・オフ

- 形式：① STEP ON
② STEP OFF

- 機能：① プログラムを1行ずつ実行するモードに変える。
② STEP 実行を解除する。

- 特徴：1 RUNの前に、STEP ONを実行するか、STOP文で停止させた後、STEP ONを実行する。
2 STEP ONモードのとき、実行はCONTで行い、RUNは無効になる。ただし、実行終了後のEND文に至った後のCONTの機能は保証できない。
3 STEP ONモードは、END文の実行、またはSTEP OFFによって解除される。
4 STEP OFFは、STEP ONモードを解除する。
5 STEP ONモードを解除した後、残りのプログラムの実行はCONTで行う。

実行例：プログラムをSTEP ONによって、ステップ・モードで実行し、STEP OFFで解除する。

```
*list
10      LET B = 3
20      FOR I = 1 TO 10
30          LET C = B^I
40          IF C > 1000 THEN
:              PRINT "TOO BIG" ; I ; C
:              END
50      NEXT I
60      END
*STEP ON
*RUN

STEP MODE
*CONT
10      LET B = 3
*CONT
20      FOR I = 1 TO 10
*CONT
30          LET C = B^I
*CONT
40          IF C > 1000 THEN
*SET A=I+5
*TYPE I;C;A;B^A
1 3 6 728.99
*STEP OFF
*CONT
TOO BIG 7 2187
END OF EXECUTION
```

参照：TRON, TROFF, SET, TYPE

APL —

TRS₁ —

CPM —

M223 ・ 本文参照。

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 ・ 本文と同じ。

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC ・ DEBUG で同じことができる。

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

文字列サーチ SEARCH

サーチ

形式：SEARCH “B”

機能：プログラム中に与えられた文字列と一致する文字列がある行をリストする。

特徴：1 変数名を変えたり，ラベル名の存在する行番号を調べるのに便利。

実行例：プログラム中で“A”のある行をリストする。

```
LIST
10 INPUT A
20 B=2*3.14*A
30 PRINT B
40 END
OK
SEARCH"A"
10 INPUT A
20 B=2*3.14*A
OK
```

参照：

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu ・本文参照。

MB16 —

IBM55 —

PC100 —

MSX —

実行プログラムの変更 CHAIN

チェイン

形式: CHAIN MERGE "TEST", 100, ALL

省略可

省略可

, DELETE 10-90

省略可

機能: ほかのプログラムをディスクよりロードして実行する。

- 特徴: 1 MERGE を付けると、今実行中のプログラムを保存する。付けないと保存しない。MERGE を付けるとき、TEST はアスキー形式^(*)でセーブしていなければならない。
- 2 100は実行開始文番号。これを省略するとプログラムの始めから実行する。
- 3 ALL を付けると変数の値を保存する。付けないと保存しない。
- 4 DELETE を付けると指定行を削除する。
- 5 メモリに入りきらない大きなプログラムを実行するのに便利。
- ^(*)SAVE "TEST", A とすればアスキー形式でセーブされる。

プログラム例: プログラム2を実行し、その途中でプログラム1をMERGE 指定をしてCHAIN する。

プログラム1

```
10 LPRINT "LINE 10"
20 LPRINT "LINE 20"
30 LPRINT "LINE 30"
50 END
```

プログラム2

```
15 LPRINT "LINE 15"
25 LPRINT "LINE 25"
35 CHAIN MERGE "TEST", 25, DELETE 35-45
45 END
```

実行例

```
LINE 15
LINE 25
LINE 25
LINE 30
```

参照: RUN, MERGE, SAVE, SWAP プログラム

APL CHAIN "TEST", 100

・変数の値は保存されない。

TRS₁ ・PC8と同じ。

CPM ・本文参照。

M223 CHAIN 1: TEST, 100

・MERGE, ALL, DELETE オプションはない。

・KILL で変数を消さないかぎり、変数は保存される。

MZK CHAIN "TEST"

・MERGE, ALL, DELETE オプションはない。

・実行はプログラムの始めから。

CBM —

PC8 RUN "TEST"

・実行はプログラムの始めから。

・変数の値は保存されない。

TRS_{II} ・PC8と同じ。

PC3 CHAIN "TEST", 100

・TEST中の最小行番号より小さなメモリ中の行番号プログラムは残る。

IF8₂ ・本文と同じ。レベル₃ ・PC8と同じ。

C180 PLOAD "TEST"

・副プログラム単位でプログラムを扱う。

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・本文と同じ。

FM8 ・PC8と同じ。

PSP CHAIN "TEST", 100

・変数の値を保存する場合、その変数をCOMMON文により宣言する。

PC88 ・本文と同じ。

N52 ・PC8と同じ。

PC6 ・PC8と同じ。

MLT ・利用者が定義した関数は保持されない。

HC ・MERGE 参照。

FP11 ・MZKと同じ。

SMC CHAIN/M/V "B: TEST", 100, DELETE10-90

・機能は本文と同じ。ただし、MERGE オプションの代わりに/Mを使用し、/Vを/Mと同じ位置に使用することによって、ALL オプションの代わりをする。

・カセット・テープでも使用できる。

MZ35 ・PC3と同じ。

X1H_u CHAIN "TEST"

・変数の値は保存される。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX —

プログラムのマージ MERGE

マージ

形式: MERGE "AFILE"

機能: メモリにあるプログラムに、指定したディスク・ファイルのプログラムを挿入する。

- 特徴: 1 指定するファイルは、アスキー形式で格納されていなければならない。そうでないときは "Bad file mode" エラーが起こる。
- 2 指定したファイルのプログラムと、メモリ中のプログラムに同じ行番号が存在するときは、ファイル中の行が優先され、メモリ中の行は削除される。
- 3 MERGE の実行が終了するとコマンド・レベルにもどる。

実行例: メモリのプログラムにファイル "PROG2" のプログラムをマージする。

```
list
10 REM sample program
20 PRINT "merge"
30 END
Ok
merge "2:prog2"
Ok
list
10 REM sample program
20 PRINT "merge"
30 PRINT "yuusen"
40 END
Ok
load "2:prog2"
Ok
list
30 PRINT "yuusen"
40 END
Ok
```

参照: CHAIN

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・APPEND "1:AFILE"

- ・指定順は、ドライブ・ナンバ、ファイル名。
- ・メモリ中のプログラムに、.BAT のファイルとして格納されているものを付けたす。
- ・APPEND するプログラムの行番号を、メモリ中の行番号より大きく整理しておくことが必要。

MZK —

CBM ・メモリ中のプログラムと、ファイル中のプログラムをつなげることとはできないが、CONCAT によって二つのファイル中のプログラムを接続できる。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 ・本文と同じ。

IF8₂ ・本文と同じ。

レベル₃ MERGE "AFILE", R

- ・R を指定すると、MERGE 終了後プログラムの先頭から実行する。

C180 ・COMBINE コマンドによって CBM と同じことが、RESIDENT 指定された IB ファイルに適用できる。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・レベル₃ と同じ。

PSP LOADS "AFILE", M

- ・M を指定すると、メモリ中のプログラムを消さずに混合しながらロードする。ソース・プログラムにエラーがあるたびに、画面にエラー・メッセージと行の内容が表示され、修正しながらロードできる。

PC88 ・本文と同じ。

N52 ・指定するファイルは、JIS 形式で格納されていなければならない。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・エラーは "BF" と出力される。また、MERGE "AFILE", R とすると MERGE 終了後はプログラム先頭から実行する。

- ・MERGE "COM0: (BLPSC)" とするとプログラムを RS-233C より読み込む場合は MERGE と同じ。BLPSC は接続条件の設定である。

FP11 MERGE "AFILE", R

- ・(, R) を指定することにより、新しく混合されたプログラムの先頭から実行される。

SMC MERGE "CTR:AFILE"

- ・装置名の指定によりディスク以外の使用も可。

MZ35 ・本文と同じ。なお、中間言語形式でもよい。

X1Hu ・本文と同じ。

オーバレイ PLOAD

ピー・ロード

形式: PLOAD "AFILE. IB / 0"

↑
必要

省略可
ドライブ番号

機能: 指定した副プログラム¹⁾を、フロッピー・ディスクから主記憶(メモリ)上へ格納する。

- 特徴: 1 IB型式²⁾のファイルは、SAVE コマンドで作成されたものであること。
 2 フォーム・ファイル³⁾の格納ができる。ただし、IBの代わりに FD を使う。
 3 PLOAD は、副プログラムの呼び出しはしない。
 4 PLOAD を連続実行すると、メモリ上のプログラムは最後に格納された副プログラムの後に格納される。
 5 ファイル名 "AFILE" は、"AFILE. IB/0" と書く。
 6 主プログラム内にはない副プログラムに対して、CALL 文を実行する前に、PLOAD 文を実行する必要がある。

用語: ¹⁾副プログラム 一般にサブルーチンは主プログラムの一部分であるが、副プログラムは、主プログラムに対して一つの独立したプログラムということができる。
²⁾IB ファイル コード・ワークと同じ内容を持ち、コード・ワークを退避することにより作成される。
³⁾フォーム・ファイル フォーム作成ユーティリティ FORM により、CRT 上に文字や、けい線などから成る表示部分、あるいは入力部分の書式を作成してできたファイル。

プログラム例: N×N 個 (#) から成る正方形を書く。副プログラムを PLOAD 文で格納し実行させる。

主プログラム

副プログラム

```

10 PLOAD "SQR.IB/0"      10 SUB SQR(P1)
20 INPUT N                20 IF P1<=0 THEN SUBEXIT
30 PRINT "N=";N           30 FOR I=1 TO P1
40 CALL SQR(N)            40 PRINT REP$( "#",P1)
50 END                    50 NEXT I
                          60 SUBEXIT
                          70 SUBEND

```

実行例

```

? 2
N=2
#####
#####

```

参照: PDELETE, CHAIN, SWAP プログラム

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ ・該当なし。ただし、MERGE "ASUB" でフロッピー・ディスク内のアスキー型式のプログラム ASUB を、主記憶上のプログラムの後に接続することができる。

PC3 ・該当なし。ただし、LOAD SUB "ASUB" で、ディスク内の中間言語型式の副プログラム ASUB を主記憶内のプログラムの後に接続する。アスキー型式の副プログラムに対しては実行できない。副プログラム内では実行できない。主記憶上のプログラムおよび全変数は保持される。

IF₈₂ ・該当なし。ただし、CHAIN MARGE "ASUB" でディスク内のアスキー型式の副プログラム ASUB を、主記憶内のプログラムの後に接続できる。

レベル₃ —

C180 ・本文参照。

M243 —

MZB ・SWAP 参照。

BUB ・該当なし。ただし、CHAIN MARGE "ASUB" ,, ALL で副プログラム ASUB を、主記憶上のプログラムの後に接続する。ALL を付けると、すべての変数が引き渡される。ALL を省略すると COMMON 文で指定した変数のみ引き渡される。

FM8 —

PSP —

PC88 ・機能は BUB と同じ。MERGE または ALL を指定しなければ DEFINT, DEFSNG, DEFDBL, DEFSTR, DEFFN, OPTION BASE, ON ERROR GOTO などの宣言文は無効になり、また ALL だけを指定しても、DEFFN, OPTION BASE, ON ERROR GOTO などは無効になる。

N52 —

PC6 —

MLT ・PC88 と同じ。

HC —

FPI1 —

SMC ・CHAIN 文で同等のことができる。

MZ35 CHAIN "AFILE, AU"

X1Hu ・該当なし。ただし、CHAIN 文はすべての変数を保存するので同等のことができる。

MB16 ・PC88 と同じ。

IBM55 ・PC88 と同じ。

オーバーレイ PLOAD

ピー・ロード

PC100 ・ PC88 と同じ.			
MSX —			

CLI モードへの復帰 GODBY

グッドバイ

形式：GODBY

機能：実行後、CLI (DOS) モードにもどる。

特徴： 1 オープン¹⁾されているファイルは、すべてクローズされる。
 2 マルチ・タスク²⁾・プログラムで END の代わりに用いる。

用語： ¹⁾オープン ファイルの使用宣言を行い、使用の準備をシステムにさせること。
²⁾マルチ・タスク 並列処理される仕事で、タスクとはジョブより小さい仕事の単位のこと。

プログラム例：1~10の数を出力した後、CLI モードへ復帰する。

```
10      FOR I = 1 TO 10
20      PRINT I
30      NEXT I
40      GODBY
NO END MARK
```

実行例

*RUN

```
1
2
3
4
5
6
7
8
9
10
```

READY

参照：BYE

APL ・ディスク・コントローラ
 の入っているスロットをSとする
 と、PR#S とすればよい。

TRS₁ CMD "S"
 ・単に DOS モードにもどるだけで
 メモリの内容は保存される。

CPM SYSTEM

M223 ・本文参照。

MZK BYE

CBM —

PC8 —

TRS₁₁ SYSTEM

PC3 BYE

IF8₂ —レベル₃ —

C180 END
 ・プログラム文中に用いることはで
 きない。

M243 ・本文と同じ。

MZB BYE

BUB —

FM8 —

PSP —

PC88 ・該当なし、これに関連し
 て BASIC モードを切り替える
 NEW ON コマンドがある。

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC ・CPM と同じ。

MZ35 BYE

X1Hu —

MB16 SYSTEM

IBM55 SYSTEM

PC100 ・同様の命令に SYSTEM
 (テキスト、変数は消去される)
 と SHELL (保存される) がある。

MSX —

CLI モードへの復帰 BYE

バーイ

形式: BYE

機能: EBASIC モードから, CLI¹⁾ (DOS²⁾) モードへもどる.

特徴: GODBY がプログラムの中で用いられるのに対して, BYE は EBASIC の直接命令であることが違う.

用語: ¹⁾CLI 入出力装置, データ・ファイルの管理, また言語のもとでのプログラムの実行を整理し, 実行される入出力管理などを効率的に整理し, そして管理するという役割を実現するための様々なコマンド (Command Line Interpreter の略).
²⁾DOS Disc Operating System. 磁気ディスク装置をコンピュータ構成内を含んでいて, それを有効に使用することを目標として設計されている OS のこと.

実行例: ファイル名 "SAMPLE" のプログラムを OLD で呼びだし, リストを出力した後 BYE で CLI モードにもどる.

```

READY
BASIC
*OLD SAMPLE
*LIST
  10      REM EBASIC MODE
  20      END
*BYE

READY

```

参照: GODBY

APL ・ディスク・コントローラ
の入っているスロットを S とし
PR#S とすればよい.TRS₁ CMD "S"
・機能は本文と同じ.CPM SYSTEM
・機能は本文と同じ.

M223 ・本文参照.

MZK ・本文と同じ.

CBM —

PC8 —

TRS₁₁ ・CPM と同じ.

PC3 ・本文と同じ.

IF8₂ —レベル₃ —C180 END
・機能は本文と同じ.

M243 ・本文と同じ.

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 ・CPM と同じ.

PC6 —

MLT ・CPM/86 下の BASIC では
SYSTEM でよい.
・スタンド・アロン M-BASIC では
用いられない.

HC —

FP11 —

SMC ・CPM と同じ.

MZ35 ・本文と同じ.

X1Hu —

MB16 ・CPM と同じ.

IBM55 ・CPM と同じ.

PC100 ・SYSTEM, SHELL 参
照.

MSX —

行の修正 EDIT

エディット

形式：EDIT 10

機能：指定した行を修正する。

- 特徴：1 指定の行番号の代わりに（.）を用いると、最後に入力した行の修正ができる。
- 2 EDIT¹⁾モードで、次のようなキーを使用して修正ができる。
 (ENTER)：コマンド・モードに無条件にもどる。n (SPACE)：カーソルを右に動かし、文字列を表示する。n, (←)：カーソルをn文字分左に動かす。(SHIFT)と(↑)：次の三つのEDITコマンドのモードからEDITモードへもどる。(X)：書き込み、(I)：文字列の挿入、(H)：文字列の削除など。
- 3 その他に、(L), (A), (R), (Q), n(D), n(C), n(S), n(K)などが修正に利用できる。(R)と(Q)はEDITコマンドのモードでは使用できない。

用語：¹⁾エディット プログラムの修正をすること、編集ともいう。

実行例：EDITを使ってプログラムの修正をする。

```

>EDIT 100
100 _
  ↓
  (L)とタイプ
  省略
  (R)(S)(I)とタイプ
100 FOR I=1 TO 10:PRINT I,I*I:NEXT I
100 FOR I=1 TO 10:PRINT _
  ↓
  (I) (I) (I) (←) (シフト) (↑)とタイプ
100 FOR I=1 TO 10:PRINT I,I*I:NEXT I
100 FOR I=1 TO 10:PRINT "I=_
  ↓
  (R)とタイプ
100 FOR I=1 TO 10:PRINT I,I*I:NEXT I
100 FOR I=1 TO 10:PRINT "I=_
>
LIST 100
100 FOR I=1 TO 10:PRINT "I=",I*I:NEXT I
  
```

参照：スクリーン・エディタ、コントロール文字

APL —

TRS_I ・本文参照。

CPM ・本文と同じ。

M223 ・EDIT 10で表示された行に対して、スクリーン・エディタ的に修正ができる。

MZK —

CBM —

PC8 —

TRS_{II} ・本文と同じ。

PC3 ・機能は本文と同じ。ただし、EDIT 70, EDIT CH, EDIT / によって連続的に何行かの修正が可能。

IF8₂ ・M223と同じ。

レベル₃ ・M223と同じ。

C180 ・SEdit /1によって起動、サブ・コマンドによって修正し、ENDの入力によって完了する。

M243 ・M223と同じ。

MZB —

BUB ・M223と同じ。

FM8 ・M223と同じ。ただし、指定行の表示の前に画面をクリアする。

PSP ・EDIT 70で指定行を表示。

PC88 ・M223と同じ。

N52 —

PC6 —

MLT ・M223と同じ。

HC —

FP11 —

SMC ・M223と同じ。

MZ35 ・PC3と同じ。なお、特徴1は(↑)キーでもできる。

X1Hu ・M223と同じ。
 ・特徴1で(.)は省略できる。

MB16 ・M223と同じ。

IBM55 ・M223と同じ。

PC100 ・M223と同じ。

MSX —

プログラム領域のクリア NEW

形式：NEW

機能：メモリの中にあるプログラムを消し、変数をすべてクリアする。

- 特徴：1 新しいプログラムを入力したいとき、コマンド・レベルで実行する。
 2 実行が終わったとき、コマンド・レベルにもどる。
 3 開かれていたファイル¹⁾はすべて閉じる。

用語：¹⁾ファイル ある目的のために集められた情報のこと。「プログラム・ファイル」と「データ・ファイル」からなる。
²⁾コード・ワーク ソース・ワークを翻訳して、実行可能な形式にした IB 形式のプログラムを格納するファイル。
³⁾ソース・ワーク ソース・プログラムを編集するために、一時的にソース・プログラムを格納するためのファイル。

実行例：メモリ中のプログラムを消去する。

```
list
10 FOR I=1 TO 10
20 PRINT "*";
30 NEXT I
40 END
Ok
new
Ok
list
Ok
```

参照：DELETE

APL ・本文と同じ。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 ・本文と同じ。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・本文と同じ。
 ・DELETE の項も参照。
 ・コード・ワーク²⁾またはソース・ワーク³⁾のファイル名指定は、BASIC 起動直後または NEW 直後に CWORK または SWORK で行う。

M243 ・本文と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 NEW ALL
 ・ALL を付けると PROG0 から PROG9 の全プログラム・エリアのプログラムと変数を消去、つけない時は現在のエリアのみ。
 ・パスワードが設定されている時には実行できない。

SMC ・本文と同じ。

MZ35 ・本文と同じ。

X1Hu ・NEW ON と CLEAR (または LIMIT) で指定した領域がクリアされる。
 INIT "0:"
 ・指定されたデバイスを初期化する。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

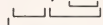
PC100^{*} ・本文と同じ。

MSX ・本文と同じ。

行番号の自動発生 AUTO

オート

形式：AUTO 70, 20



省略可

機能：1 行入力終了のたびに、自動的に行番号を発生する。

- 特徴：1 AUTOの指定順は、プログラムの最初の行番号、(,), プログラムの増分で、省略したときはそれぞれ10になる。ただし、増分のみを省略したときは直前の AUTO コマンドの増分が適用される。
- 2 AUTOは(CTL)-(C), または (STOP) キーの入力によって終了する。このときの行は格納されない。
- 3 (CTL)-(C) の入力後はコマンド・レベル¹⁾にもどる。
- 4 プログラムを入力するとき便利である。

用語：¹⁾コマンド・レベル プログラムの入力や修正、実行ができる状態。あるいはダイレクト・コマンドが使える状態。

実行例：行番号を20から増分5で自動的に発生する。

```
AUTO 20,5
20 FOR I=1 TO 10
25 PRINT "*";
30 NEXT I
35 END
40
OK
```

参照：

APL —

TRS₁ ・書式は本文と同じ。ただし、最初の行番号を指定しないと0行から、(.)を指定すると現在の行番号+10より行番号を生成する。

CPM ・本文と同じ。ただし、発生した行がプログラム中に既にあるときは(*)が表示され、(RETURN)キーを入力したときは、古い行が、文字を入力した後 (RETURN) キーを入力したときは新しい行が優先する。

M223 ・最初の行番号は省略できない。増分は省略すると10ずつとられる。

・解除は (CR) を2度つづけて入力する。

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・TRS₁と同じ。ただし、CPMと同じ(*)の機能がある。

PC3 ・本文と同じ。ただし、解除には (CL) キー、(AUTO), (ENTER) キー、または (HOME) キーを用いる。

IF8₂ ・CPMと同じ。

レベル₃ ・本文と同じ。

C180 AUTO 70

・指定は、最初の行番号で、増分10で表示する。省略するとメモリ中の最大行番号+10行から増分10で表示する。

- ・解除はキャンセル・キーによる。
- ・LIST コマンドが実行されるまでは、前に実行した AUTO が無効にならない。
- ・入力されている番号より小さな番号をAUTOで指定することはできない。

M243 ・M223と同じ。

MZB ・本文と同じ。

BUB ・CPMと同じ。

FM8 ・すでにある番号は入力できない。(RETURN)キーで終了する。

PSP ・CPMと同じ。

PC88 ・CPMと同じ。

- N52 ・行番号のみが指定されたときの増分値は10。
- ・(,)と増分値だけが指定されたときの行番号は0である。
 - ・特徴1において増分値のみを省略し、以前にまだAUTOコマンドが使われていなければ、次に入力するプログラムにだけ行番号がつけられ、AUTO コマンドの処理は終わる。
 - ・AUTOはストップ・キーを押すまで有効で、その後コマンド・レベルにもどる。

PC6 —

MLT ・本文と同じ。

HC ・特徴②の (STOP) キーは (BREAK) キーである。

FP11 ・増分を省略すると10になる。

行番号の自動発生 AUTO

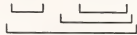
オート

<ul style="list-style-type: none"> ・解除にはプログラムを入力せずに RETURN を押す。 ・発生した行が既に存在する時、♥を表示して発生を中止しコマンド入力待ちとなる。 			
<p>SMC ・増分を省略したときは10になる。</p> <ul style="list-style-type: none"> ・発生した行が既にある時はその行が表示される。 ・中間行も表示する。 			
<p>MZ35 ・増分を省略すると増分は10になる。</p> <ul style="list-style-type: none"> ・オート機能を解除するには CL キーを押すか、または ☐ キーを操作してライン・ナンバを消した後 AUTO ENTER とする。あるいは CTL-HOME キーを用いる。 			
<p>X1Hu ・本文と同じ。</p> <ul style="list-style-type: none"> ・(.) を用いることができる。 			
<p>MB16 ・CPM と同じ。</p>			
<p>IBM55 ・TRS₁ と同じ。ただし、解除には BREAK キーを用いる。</p>			
<p>PC100 ・CPM と同じ。</p>			
<p>MSX ・CPM と同じ。</p>			

プログラムのリスト LIST

リスト

形式: LIST 70 -170



省略可

機能: メモリの中にあるプログラムの全部、または一部をディスプレイに出力する。

- 特徴:
- 1 実行が終了すると、コマンド・レベルにもどる。
 - 2 すべて省略したときは、プログラムのすべてを出力する。ただし、途中で実行を終了するときは、(STOP) キーの入力による。
 - 3 最初の行番号のみ指定したときは、その行のみ出力する。
 - 4 (一)まで指定したときは、指定した行以降すべての行を出力する。
 - 5 (一)と2番目の行番号を指定したときは、プログラムの最初の行から、指定した行までを出力する。
 - 6 すべて指定したときは、前に指定した行番号から、後に指定した行番号までを出力する。
 - 7 (一)の代わりに(,)を用いてもよい。

実行例: プログラム・リストを出力してみる。

```
list
10 INPUT "A=";A
20 IF A = 0 THEN GOTO 50
30 PRINT A;"^2=";A^2
40 GOTO 10
Ok
list 20-30
20 IF A = 0 THEN GOTO 50
30 PRINT A;"^2=";A^2
Ok
list -20
10 INPUT "A=";A
20 IF A = 0 THEN GOTO 50
```

参照: LLIST

APL LIST 70, 170

TRS_I ・本文と同じ。(一)の代わりに(,)は使えない。CPM ・TRS_Iと同じ。

M223 LIST 70, 5, 170

・70行から170行まで、5行ずつ表示する。最初の行番号のみ指定したときは、その行以降すべて出力する。5を省略したときは23行ずつ表示。続行はキーの入力による。(LF) キーを入力すると1行ずつ、(H) キーを入力すると11行ずつ表示する。コマンド・レベルにもどるには、(ESC) キーの入力による。

・オペランドの指定によりファイルやプリンタへの出力も可。

MZK ・本文と同じ。

CBM ・TRS_Iと同じ。

PC8 ・本文参照。

TRS_{II} ・TRS_Iと同じ。

PC3 ・APLと同じ。ただし、一行のみ表示したいときは、LIST 70, 70を使用、LIST 70とすると70行以降のすべての行を表示する。

IF8₂ ・TRS_Iと同じ。レベル₃ ・本文と同じ。

C180 LIST 70, 170, \$LP
 ・\$LPを指定しないとAPLと同じ。
 ・\$LPを指定するとプリンタに出力される。

M243 ・M223と同じ。

MZB ・LIST, LIST-70のみ使用可。

・リスト中 (SPACE) バーで表示を一時中止、(BREAK) キーで中止できる。

BUB ・TRS_Iと同じ。

FM8 ・本文と同じ。ただし、エラーが起きたとき、LISTでエラーの起きた文の内容を表示できる。
 ・表示の一時中止は (ESC) キー。

PSP ・APLと同じ。ただし、画面に23行ずつ表示して停止、このとき続けなければ、(RETURN) キーを、コマンド・レベルにもどりたいときは、それ以外のキーを入力する。

PC88 ・TRS_Iと同じ。ただし、(*)+ラベルを行番号の代わりに使える。

N52 ・途中での実行の中止には (中断) キーを押すか、あるいは (CTL) キーと (中断) キーを同時に押す。
 ・プログラム実行中にエラーが起きたときにエラーの起きた文の内容をLISTで表示できる。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・出力中は (PAUSE) で一時停止でき、(BREAK) で中断できる。
 ・LIST "COM0:" でRS-232Cポートへプログラム・リストを出力する。

FP11 ・TRS_{II}と同じ。ただし、

プログラムのリスト LIST

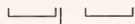
リスト

<p>(,) は BASIC が最後に扱った行の番号を示す。</p> <ul style="list-style-type: none"> ・ STOP キーで中断。 ・ CONT キーで継続。 ・ パスワードが設定されているときはリストがとれない。 <hr/> <p>SMC ・ PC88 と同じ。ただし、(,) は使用不可。SAVE/P でセーブされたプログラムはリストがとれない。</p> <hr/> <p>MZ35 ・ PC3 と同じ。</p> <hr/> <p>X1Hu ・ (-) の代わりに (,) は使えない。</p> <ul style="list-style-type: none"> ・ LIST “TEST” でファイルのリストがとれる。 <hr/> <p>MB16 ・ TRS₁ と同じ。また、LIST 行一行、“ファイル指定”にて各装置にリストできる。</p> <hr/> <p>IBM55 ・ MB16 と同じ。</p> <hr/> <p>PC100 ・ PC88 と同じ。ただし、ラベルには*がなくてもよい。</p> <hr/> <p>MSX ・ TRS₁ と同じ。</p> <hr/>			
--	--	--	--

リスト印刷 LLIST

エル・リスト

形式: LLIST 100-150



省略可

機能: プログラムをプリンタに出力する。

- 特徴: 1 行番号が省略されると、メモリ上のプログラム・リストをすべてプリンタに打ち出す。
- 2 初めの引数で、開始行番号、次の引数で、停止行番号を指定でき、その範囲だけプリンタに打ち出す。
- 3 132 文字の幅のプリンタを仮定して、このコマンドの実行が終わるとコマンド・レベルにもどる。
- 4 プログラム中でも使えるが、そのときはこのコマンドを実行して止まる。また、このとき引数は使えない。
- 5 行番号の指定方法は LIST と同じ。(一)の代わりに(,)も使える。

用語: ¹⁾スロット 拡張ボードを差し込むパソコン・ボード上の差し込み口。

プログラム例: プログラムのリストをプリンタに出力する (プログラム中で用いた例)。

```
10 REM *** TEST PROGRAM          ***
20 REM *** FOR LLIST COMMAND ***
30 LLIST:END
```

実行例

```
10 REM *** TEST PROGRAM          ***
20 REM *** FOR LLIST COMMAND ***
30 LLIST:END
```

参照: LIST

APL ¹⁾スロット #2 にインターフェース・カードがある場合 PR#2とし、LIST、中止は PR#0。

TRS_I ¹⁾(-)の代わりに(,)は不可。

CPM ¹⁾TRS_Iと同じ。

M223 LIST 'P'OUT, 3, 100, 2, 150
・第1の引数でプリンタのモードを指定する。開始行と停止行との間に意味のない値(≠0)を入れる。

MZK LIST/P 100, 150
・機能は本文と同じ。

CBM LIST 100-200
・この実行の前に OPENn, 4 で、論理番号 n のファイルを、デバイス番号 4 のプリンタにオープンして、CMD4 でプリンタに制御を移す。
・これらは、プログラムでも実行できる。ほかは本文と同じ。

PC8 ¹⁾本文参照。

TRS_{II} ¹⁾TRS_Iと同じ。

PC3 LIST PRINT 100, 150
・機能は本文と同じ。ただし、CSAVE@でセーブしたプログラムは出力できない。

IF8₂ ¹⁾TRS_Iと同じ。

レベル₃ LIST "LPT0:", 100-150
・機能は本文と同じ。引用符で囲んだプリンタ名は LPT0: ~ 2: がある。行番号間はコンマ(,)を使ってもよい。

C180 LIST 100, 150, \$LP

・機能は本文と同じ。

M243 ¹⁾M223と同じ。

MZB ¹⁾MZKと同じ。

BUB ¹⁾TRS_Iと同じ。

FM8 LIST "LPT0:", 100-200
・"LPT0:" でプリンタ LPT0: に出力することを示す。ほかは本文と同じ。

PSP LIST# 50, 100
・機能は本文と同じ。

PC88 ¹⁾TRS_Iと同じ。ただし、(*)+ラベル名を行番号の代わりに使える。

N52 ¹⁾プログラム・リストがプリンタに出力されることを除いて、コマンドの使い方は N52 の「LIST」コマンドと同じである。

PC6 ¹⁾本文と同じ。

MLT ¹⁾本文と同じ。

HC ¹⁾本文と同じ。

FP11 ¹⁾LIST の項も参照。

SMC ¹⁾PC88と同じ。LIST の項も参照。

MZ35 ¹⁾PC3と同じ。

X1Hu ¹⁾本文と同じ。ただし、(,)は使えない。

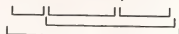
MB16 ¹⁾TRS_Iと同じ。

IBM55 ¹⁾TRS_Iと同じ。

行番号の整理 RENUM

リナンバー

形式: RENUM 70, 170, 20



省略可

機能: プログラムの行番号を整理して付け替える。

- 特徴: 1 RENUM の指定は、新しく付ける最初の行番号(), 付け替えを始める現在のプログラムの行番号、新しく付ける行番号の増分の順で指定する。
- 2 RENUM は、GOTO, GOSUB, THEN, ON~GOTO, ON~GOSUB, ERL など参照する行番号も新しい行番号に即してつけなおされる。参照する行番号が存在しないときは、"Undefined line xxxx in yyyy" のエラーが出力される。このとき xxxx は RENUM に影響されない。
- 3 RENUM は、プログラムの行の順序を変更するのには使用できない。
- 4 RENUM は、65529 以上の行番号を発生できない。このとき "Illegal function call" エラーが起こる。
- 5 RENUM のみだと行番号 10 から 10 きざみとなる。

実行例: 行番号 1 からのプログラムを、行番号 10 から増分 20 で整理する。

```
list
1 FOR I=1 TO 10
15 PRINT "*" ;
100 NEXT I
150 END
Ok
renum 10,1,20
Ok
list
10 FOR I=1 TO 10
20 PRINT "*" ;
100 NEXT I
200 END
```

参照: NO LIST, KEY LIST

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 RENUM 70 T 170, 5

- ・指定は、整理したいプログラムの最初の行番号、T、新しい最初の行番号、(), 行番号の増分、の順。その他は本文と同じ。

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 REN 70, 10

- ・最初の行番号 70, 増分 10 でプログラムを整理する。旧番号は指定できない。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 RENUMBER 70, 20

- ・機能は PC3 と同じ。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・行番号は 0 ~ 63999。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・特徴 2 で yyyy は RENUM

に変更される。

- ・RENUM は 65000 より大きい行番号を発生することはできない。

PC6 RENUM 1000, 100

- ・本文と同じだが、増分は 10 に固定される。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・行番号の最大は 64999。

SMC ・ラベルの使用も可。

MZ35 REN 70, 170, 20

- ・特徴 4 で 99999 まで使え、特徴 5 はエラー。

- X1Hu ・未定義行番号は 65535 になる。
- ・エラー・メッセージは出ない。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

行の削除 DELETE

デリート

形式：DELETE 70-170

省略可

パラメータ

機能：プログラムの行を削除する。

- 特徴：
- 1 DELETE の指定は、削除したい最初の行番号、(,) または (-), 削除したい最後の行番号の順。
 - 2 最初の行番号だけを指定したときは、その行のみを削除する。
 - 3 最初の行番号を省略したときは、指定した行番号までのすべての行を削除する。
 - 4 1 行のみの削除は、行番号だけを入力すればよいが、連続した何行かを削除したいとき使用すると便利である。

実行例：プログラムの30行から40行までを削除する。

```

10 FOR I=1 TO 10
20 PRINT "SQRT(";I;")=";SQR(I)
30 BEEP 1
40 BEEP 0
50 NEXT I
Ok
DELETE 30-40
Ok
LIST
10 FOR I=1 TO 10
20 PRINT "SQRT(";I;")=";SQR(I)
50 NEXT I

```

参照：NEW

APL DEL 70, 170
・パラメータは省略できない。

TRS_i ・(,)の使用は不可。

CPM ・TRS_i と同じ。

M223 70, 170
・70-で70行以降すべてを削除できる。

MZK —

CBM —

PC8 ・本文参照。

TRS_{ii} ・TRS_i と同じ。ただし、
後の行番号に(,)を使用すると、
プログラムの最後の行を指定した
ことになる。

PC3 ・(-)の使用は不可。

IF8₂ ・TRS_i と同じ。
・現在の番号を(,)で指定できる。

レベル₃ ・本文と同じ。

C180 DELETE 70, 170, *
・指定順は、削除したい最初の行番
号、最後の行番号、(*)。
・*を指定すると、デバッグ行のみ
削除、*と行番号の両方を省略し
てはいけない。
・PLOAD で格納されている副プロ
グラムの消去は PDELETE
"AFILE" を用いる。

M243 ・M223 と同じ。

MZB ・本文と同じ。

BUB ・TRS_i と同じ。

FM8 ・(-)または(,)のみを指
定すると、すべての行が削除され
る。

PSP ・(-)の使用は不可。D 70,
で70行以降が消去できる。

PC88 ・TRS_i と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・TRS_i と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・TRS_i と同じ。ただし、
最後の行番号のみを省略すると最
初の行番号から終わりまで。

MZ35 ・PC3 と同じ。ただし、特
徴2, 3は使えない。

X1Hu ・DELETE, で全部削除
される。

MB16 ・TRS_i と同じ。

IBM55 ・TRS_i と同じ。

PC100 ・TRS_i と同じ。

MSX ・TRS_i と同じ。

機密の保持 NO LIST

ノー・リスト

形式: NO LIST 70, 170

機能: プログラムの機密保持のため, プログラムのリストができないようにする.

- 特徴: 1 NO LIST の指定は, ロック¹⁾したいプログラムの最初の行番号, (,), 最後の行番号である.
- 2 RENUM した後, NO LIST, 再び RENUM の実行, KEY の実行で一回の操作になる. このとき, KEY の指定は単精度の数値データで, これが暗証番号 (KEY) になる.
- 3 プログラムがロックされた時 LOCK! を表示する.
- 4 ロックの解除にも KEY コマンドを使う.
- 5 プログラムの機密保持のために便利である.

用語: ¹⁾ロック 上記の機能を果たすことをロックする, ともいう.

実行例: プログラムに NO LIST でロックをかけ, 次に KEY によってロックを解除する.

```
*LIST
10      INPUT "A=" ; A
20      IF A = 18 OR A = 8 THEN GOTO 50
30      PRINT "FALSE !"
40      GOTO 10
50      PRINT "TRUE !"
60      GOTO 10
70      END

*RENUM
*NO LIST 20,60
*KEY 123
LOCK !

*LIST
10      INPUT "A=" ; A
20      IF A = 18 OR A = 8 THEN GOTO 50
30      PRINT "FALSE !"
40      GOTO 10
50      PRINT "TRUE !"
60      GOTO 10
70      END
```

参照: KEY ロック

APL —

TRS_I —

CPM —

M223 ・本文参照.

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 ・CSAVE@でセーブすれば機密が保持される.

IF8₂ —レベル₃ —

C180 —

M243 ・本文と同じ.

MZB —

BUB —

FM8 UN LIST 70

- ・指定した行以降をリストできない.
- ・UN LIST 実行後SAVEするとき
は, バイナリ型式による.
- ・解除する KEY に相当するコマンドはない. ただし,
POKE &H1E7, &HFF
POKE &H1E8, &HFF
で行える.

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 ・PASS が NO LIST と
KEY の両方の機能をあわせもつよ
うな働きをする.

SMC ・SAVE/P でリストをとれ
なくすることができる.

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

キーの指定と解除 KEY ロック

形式: KEY 2

機能: 暗証番号 (キー) の指定と NO LIST で行ったロックの解除。

- 特徴: 1 RENUM, NO LIST, RENUM, KEY の操作中, KEY の後に書かれた単精度の数値データが, 暗証番号として指定される。
- 2 プログラムの機密を保持する上で, NO LIST は有効であるが, 暗証番号 (KEY) を知っている者は, KEY の後にこの数値データを指定することで, NO LIST で行ったロックを解除できる。

実行例: プログラムに NO LIST でロックをかけ, 次に KEY でロックを解除する。

```
*LIST
10 INPUT "A=" ; A
20 IF A = 18 OR A = 8 THEN GOTO 50
30 PRINT "FALSE !"
40 GOTO 10
50 PRINT "TRUE !"
60 GOTO 10
70 END

*RENUM
*NO LIST 20,60
*KEY 123
LOCK !
*LIST
10 INPUT "A=" ; A
20 END

*KEY 789
NO !
*KEY 123
*LIST
10 INPUT "A=" ; A
20 IF A = 18 OR A = 8 THEN GOTO 50
30 PRINT "FALSE !"
40 GOTO 10
50 PRINT "TRUE !"
60 GOTO 10
70 END
```

参照: NO LIST

APL —

TRS_I —

CPM —

M223 ・本文参照。

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —レベル₃ —

C180 —

M243 ・本文と同じ。

MZB —

BUB —

FM8 POKE &H1E7, &HFF
POKE &H1E8, &HFF
でやる。

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FPI1 PASS ALL "CASIO"

・NO LIST と KEY を一つの命令で実行。

・パスワードの設定および解除をする。

・ALL を付けると PROG0 から PROG9 までの全プログラム・エリアが保護される。ない場合現在のエリアのみ。

・パスワードは 1 から 8 の文字定数。右端の (") は省略できない。

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

キーボード入力のシミュレート KEY 0

キー・ゼロ

形式：KEY0, "LIST" + CHR\$ (13)

機能：プログラム中でキーボードからの入力をシミュレートできる。

特徴： 1 入力できる文字列は63文字まで。
2 プログラム中では、INKEY\$ などの入力命令で実行される。

実行例：キーボードから入力する文をシミュレートさせる。

```
KEY0."FOR I=1 TO 3:PRINT I::NEXT I"+CHR$(13)
OK
FOR I=1 TO 3:PRINT I::NEXT I
1 2 3
OK
```

参照：

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF 8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu ・本文参照。

MB16 —

IBM55 —

PC100 —

MSX —

プログラム・エリアの切り替え LOGIN

ログイン

形式：LOGIN 3, R

プログラム・ノ 省略可
エリア番号

機能：プログラム・エリアを切り替える。

- 特徴：
- 1 プログラム・エリアが五つに分割してあるので、それぞれに別のプログラムが収納できる。
 - 2 エリアは1から5までである。
 - 3 NEW, LIST, LOAD, SAVE は、その時 LOGIN されているエリアだけに有効である。
 - 4 R オプション指定により、プログラム・エリアを切り替えたあと、そのプログラムを実行する。
 - 5 プログラム名をつけるには、TITLE プログラム名を実行する。
 - 6 プログラム・エリアの使用状態の表示は STAT により行う。

実行例：プログラム・エリアを1から2へ切り替えて、異なるプログラムが入っていることを確める。

```
LOGIN 1
P1:      183 Bytes
LOGIN 2
P2:      94 Bytes
```

参照：PCOPY

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC — 本文参照。

FP11 PROG 3

- ・プログラム・エリア番号は0から9である。
- ・R オプション指定はなく、実行後コマンド待ちとなる。
- ・SYSTEMにより各プログラム・エリアの状態が表示される。

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

ほかのプログラム・エリアにプログラムをコピー PCOPY

ピー・コピー

形式：PCOPY 3

プログラム・エリア番号

機能：BASIC のプログラムをほかのプログラム・エリアにコピーする。

- 特徴：1 プログラム・エリアが五つに分割されているので、一つのエリアからほかのエリアにプログラムをコピーできる。
2 指定されたエリアにプログラムが存在していたり、現在ログインされているプログラム・エリアにプログラムがない場合にはエラーとなる。

実行例：プログラム・エリア1に入っているプログラムを2へコピーし、リストにより確める。

```
100 REM***** PCOPY TEST
PROGRAM *****
110 REM コマンドプログラム PROGR
AMITIA 1 ニ
120 REM イレ、PROGRAMITIA 2
カ アイテムコトヲ
130 REM タシカメ、PCOPY 2 ヲシ
ツクワシ、コヒニ
140 REM タシカメ、タシカメニ
STAT 2
P2: 0 Bytes
COPY
PCOPY 2
COPY

LOGIN 2
P2: 157 Byte
COPY
LIST
100 REM***** PCOPY TEST
PROGRAM *****
110 REM コマンドプログラム PROGR
AMITIA 1 ニ
120 REM イレ、PROGRAMITIA 2
カ アイテムコトヲ
130 REM タシカメ、PCOPY 2 ヲシ
ツクワシ、コヒニ
140 REM タシカメ、タシカメニ
```

参照：LOGIN, STAT

APL —

TRS —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS —

PC3 —

IF8 —

レベル —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC

・本文参照。

FP11 MOVE PROG3

・プログラム・エリア1に入っているプログラムを2へコピーし、リストにより確める。

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

注釈文 REM

形式：REM BASIC

注釈文

機能：プログラム中に注釈文を入れるのに使う。

- 特徴：1 注釈文の行には、複数命令(:)は書けないので注意。
 2 注釈文を適当に入れておくと、プログラムのデバッグがやりやすくなる。
 3 REM の代わりにセミクォーテーション(')を使用できる機能もある(シンボルの項参照)。

プログラム例：REM, (')を用いてプログラム中に注釈を入れる。

```
10 REM -----BASIC-----
20 ' -----PROGRAM-----
30 I=3
40 LPRINT "I=", I
50 END
```

実行例

I=

3

参照：シンボル

APL ・本文と同じ。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・(/)と(!)が REM の代わりに使用可。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 ・(!)を REM の代わりに使用できる。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・PC3 と同じ。

M243 ・M223 と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・(')のほか(*)も REM の代わりに使用できる。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・REM 〇のように空白が必要。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・複数命令も可。

MZ35 ・PC3 と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

停止 STOP

ストップ

形式：STOP

機能：実行中のプログラムを停止し、コマンド・レベルにもどる。

特徴： 1 プログラム中のどこで使用してもよい。
 2 STOP 文が実行されると次のメッセージが表示される。BREAK IN nnnn (行番号)
 3 STOP 文はファイルをクローズ¹⁾しない。
 4 CONT コマンドにより、プログラムの実行が再開される。
 5 プログラムのデバッグ時にプログラム中のポイントになる場所に PRINT 文などを入れておき、少しずつプログラムを実行していくと便利である。

用語： ¹⁾クローズ オープンされたファイルに対して、読み書きが終了したことを知らせること。

プログラム例：行番号30にSTOP文をおき、プログラムの実行をそこで一度停止させる。停止後、CONTを入力しプログラムを続行する。

```
10 I=3
20 PRINT I
30 STOP
40 I=I*2
50 PRINT I
60 END
```

実行例

```
run
3
Break in 30
Ok
cont
6
Ok
```

参照：CONT, END

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ 本文と同じ。ただし、表示するメッセージは、STOP AT THE LINE NO. nnnn

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS₁₁ ・ 本文と同じ。

PC3 ・ 本文と同じ。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。ただし、表示するメッセージは、E100である。

M243 ・ M223 と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。ただし、メッセージは表示されない。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ メッセージは、Stop PROG n (プログラム・エリア番号) in nnnn (行番号)。

SMC ・ 本文と同じ。

MZ35 ・ PSP と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。

MSX ・ 本文と同じ。

プログラムの再実行 CONT

コンティニュー

形式：CONT

機能：(STOP) キーの押し下げ、または STOP, END 文の実行によって停止していたプログラムの再実行。

- 特徴：1 中断された文、または STOP 文、END 文の次の文からプログラムの実行が継続する。
 2 INPUT 文からのプロンプト¹⁾文を出力した後、停止したときは、プロンプト文を再びプリントするところから実行を再開する。
 3 プログラムの停止中に、プログラムに変更を加えたとき CONT は無効になる。
 4 CONT は、普通 STOP 文とともに用い、プログラムの実行停止中に、ダイレクト・モード・ステートメントによって中間結果を調べたり、変更することができ、デバッグに便利である。

用語：¹⁾プロンプト キーボード入力を受け入れる準備ができていないことを示すために、コンピュータから出される文字列。

実行例：一時停止したプログラムを、CONTで次の文から再実行する。

```
run
A=? 11
Break in 20
Ok
?a
11
Ok
cont
A squared is 121 .
Ok
```

参照：STOP, END, GOTO

APL ・本文と同じ。

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 CONT n or s
 ・ライン・ナンバ n または、ライン・ラベル s から実行を再開する。
 ・CONTの後に (/) を指定すると 1 行のみ実行する。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・CONTの代わりに (ENT) キーを入力する。

M243 ・本文と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。ただし、INPUT、または GPCURSOR 文で実行を停止したときは、その文の始めから実行する。

PSP ・本文と同じ。ただし、プログラムの停止は (CTRL) キーと (STOP) キーを同時に入力するか、STOP 文の実行による。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・プリンタ、カセットの入出力時の場合には、CONTによる実行再開はできない。

FP11 ・本文と同じ。

SMC ・END 文については不可。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。ただし、プログラムの中断は (SHIFT) + (BREAK)。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

プログラムの終了 END

エンド

形式：END

機能：実行中のプログラムを終了し、すべてのファイルを閉じた後に、コマンド・レベルにもどる。

特徴：1 END 文はプログラムの最後にある必要はなく、複数個使用してもよいので、プログラム開発時に、サブルーチン¹⁾の前などに一文入れておくと、プログラムの暴走をふせげる。

用語：¹⁾サブルーチン プログラム中で繰り返して行う部分を一つのプログラムとして扱い、必要な時、そのプログラムを何度でも実行できるようにしたもの。

プログラム例：プログラム中に END は複数個あってもよい。

```
10 I=3
20 ON I-2 GOTO 40,70
30 END
40 LPRINT "LINE NO. 40 I-2=";I-2
50 I=4
60 GOTO 20
70 LPRINT "LINE NO. 70 I-2=";I-2
80 END
```

実行例

```
LINE NO. 40 I-2= 1
LINE NO. 70 I-2= 2
```

APL ・ 本文と同じ。

TRS_I ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ 最終ステートメントに
END がないと NO END MARK
が表示される。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 主プログラムの最後に一
つだけ書く。

IF8₂ ・ 本文と同じ。

レベル₃ ・ 本文と同じ。

C180 ・ PC3 と同じ。

M243 ・ M223 と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FPII ・ 本文と同じ。

SMC ・ END 文を実行すると、
* End* と表示される。

MZ35 ・ 本文と同じ。

XIHu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。

MSX ・ 本文と同じ。

参照：STOP

代入文 LET

形式: LET A=B+C*2

省略可 変数名 式

機能: 等号の右の式で表される値を、等号の左の変数に代入する。

- 特徴: 1 等号の左右で数値変数の型が違う場合、左の変数名によって宣言された型に変換される。
 2 等号の左右は、文字変数または数値変数に統一されなければエラーとなる。"TYPE mismatch".
 3 LETは多くの場合省略する。

プログラム例: I*2+JをIに代入する。

```

10 I=3:J=5
20 GOSUB 60
30 LET I=I*2+J
40 LPRINT "I=",I
50 END
60 LPRINT "SUB 60 I=",I
70 RETURN

```

実行例

```

SUB 60 I=      3
I=          11

```

参照: 算術演算子

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ 数値定数の場合、等号の左右は同じ精度にしなければいけない。
 ・ ダイレクト・コマンドとして実行させるには、SET A=10を用いる。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 本文と同じ。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 LET A=B+C*2
 ・ LETの後に空白が必要。
 ・ 数値を文字化してレコード変数、アイテム変数へ代入するには、MOVE ES=A+3〔2〕を用いる。

M243 ・ M223と同じ

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。

SMC ・ 本文と同じ。

MZ35 ・ 本文と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。

MSX ・ 本文と同じ。

データ文からの入力 READ

リード

形式: READ A, E\$, B (5)

*省略・繰り返し可

機能: DATA 文より値を読み出し、変数に入力する。

- 特徴: 1 READ 文は、いつも DATA 文と組み合わせて使わなければならない。
- 2 READ 文の変数は、数値変数でも文字変数でもよいが、それぞれ対応する変数の型と DATA 文の型が一致していなければならない。
- 3 一つの READ 文が複数の DATA 文を参照したり (この場合は順番に参照される)、複数の READ 文が一つの DATA 文を参照したりすることができる。

プログラム例: READ 文, DATA 文, RESTORE 文を組み合わせた例。

```

10 DIM A(5)
20 FOR I=1 TO 5
30 READ A(I)
40 LPRINT A(I);
50 NEXT I:LPRINT
60 RESTORE 100
70 READ A1
80 LPRINT A1
90 DATA 1,2,3,4
100 DATA 100
110 END

```

実行例

```

1 2 3 4 100
100

```

参照: DATA, RESTORE

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ 本文と同じ。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS₁₁ ・ 本文と同じ。

PC3 ・ 本文と同じ。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。

M243 ・ 本文と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 特徴 3 はない。

FP11 ・ 本文と同じ。

SMC ・ 本文と同じ。

MZ35 ・ 本文と同じ。

X1Hu

DTL

・ 現在読んでいる DATA 文の行番号を与える。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。

MSX ・ 本文と同じ。

データ文 DATA

データ

形式: DATA 100, 200, タテ

*省略・繰り返し可

機能: READ 文で読み出される数値, および文字定数を格納する.

- 特徴:
- 1 DATA 文は, プログラム中のどこに置いてもよい.
 - 2 一つの DATA 文は, 1 行に入るだけの定数を入れることができる.
 - 3 DATA 文には固定小数点, 浮動小数点または整数を含んでもよいが, 数式は許されない.
 - 4 DATA 文に含まれる文字定数は, コンマ(,), セミコロン(;) または前後に意味のある空白を含む場合のみ引用符(") で区切り, それ以外の場合は引用符で区切る必要はない.
 - 5 大量のデータがあるとき使うと, キーボードから入力する必要がないので便利である.

プログラム例: READ 文, DATA 文, RESTORE 文を組み合わせ例.

```

10 READ A1,A2,E1$,E2$
20 LPRINT A1;A2;TAB(5);E1$;E2$
30 RESTORE 70
40 READ E1$,E2$
50 LPRINT E1$;E2$
60 DATA 10,100
70 DATA タテ," ヨヨ"
80 END

```

実行例

```

10 100 タテ ヨヨ
タテ ヨヨ

```

参照: READ, RESTORE

APL ・本文と同じ.

TRS₁ ・本文と同じ.

CPM ・本文と同じ.

M223 ・本文と同じ. ただし, 文字列は引用符(") で囲む.

MZK ・本文と同じ.

CBM ・本文と同じ.

PC8 ・本文参照.

TRS_{II} ・本文と同じ.

PC3 ・本文と同じ.

IF8₂ ・本文と同じ.レベル₃ ・本文と同じ.

C180 ・本文と同じ. ただし, 2進数値のときセミクォテーション(') で囲まなくてよい.

M243 ・M223 と同じ.

MZB ・本文と同じ.

BUB ・本文と同じ.

FM8 ・本文と同じ.

PSP ・本文と同じ.

PC88 ・本文と同じ.

N52 ・本文と同じ.

PC6 ・本文と同じ.

MLT ・本文と同じ.

HC ・本文と同じ.

FP11 ・本文と同じ.

SMC
い. ・セミコロンは使用できない.

MZ35 ・M223 と同じ.

X1Hu ・本文と同じ.

MB16 ・本文と同じ.

IBM55 ・本文と同じ.

PC100 ・本文と同じ.

MSX ・本文と同じ.

読み出し位置の指定 RESTORE

リストア

形式：RESTORE 100

省略可

機能：DATA 文を始め（指定した行番号）から読めるようにする。

- 特徴：1 RESTORE 文が実行されると、その次の READ 文はプログラム中の最初の DATA 文から読み出す。
 2 行番号を指定すると、その次の READ 文は指定された行番号の DATA 文から読み出す。
 3 同じ DATA 文を何度も読んだり、複数の READ 文が一つの DATA 文を参照したりする時に使うと便利である。

プログラム例：READ 文、DATA 文、RESTORE 文を組み合わせた例。

```
10 DIM A(5)
20 FOR I=1 TO 5
30 READ A(I)
40 LPRINT A(I);
50 NEXT I:LPRINT
60 RESTORE 100
70 READ A1
80 LPRINT A1
90 DATA 1,2,3,4
100 DATA 100
110 END
```

実行例

```
1 2 3 4 100
100
```

参照：READ, DATA

APL ・本文と同じ。

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・本文と同じ。

M243 ・本文と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・RES と短縮可。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・本文と同じ。

MZ35 ・本文と同じ。

X1Hu ・行番号の代わりにラベル名も使える。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

スワップ

値の交換 SWAP_{VALUE}

形式：SWAP A1, A2

機能：A1 と A2 の値を入れ替える。

- 特徴：1 どの型の変数でも交換可能である（整数、単精度、倍精度、ストリング、配列変数）。ただし、その二つの変数の型が一致している必要がある。
- 2 変数を交換するとき、別の変数に置き替える必要がないので便利である。

プログラム例：“ADAM” と “EVE” の文字列を入れ替える。

```

10 A$="ADAM":B$=" AND ":C$="EVE"
20 LPRINT A$;B$;C$
30 SWAP A$,C$
40 LPRINT A$;B$;C$
50 END

```

実行例

```

ADAM AND EVE
EVE AND ADAM

```

参照：

APL —

TRS_I —

CPM ・本文と同じ。

M223 —

MZK ・該当なし、MZK の
SWAP とは異なるものである。

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 —

M243 —

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・双方が数値式で型が異なる時、型の自動変換が行われる。

SMC ・本文と同じ。

MZ35 —

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

暗黙の整数型指定 DEFINT

デファイン・インテジャ

形式: DEFINT I-N, S-T

省略可・省略・繰り返し可

機能: 変数の型を整数型に定義する。

- 特徴:
- 1 指定文字で始まる変数を整数に指定する。指定文字は最初の文字のみ。
 - 2 変数毎の型宣言文字が常にこの指定より優先する。
 - 3 特にこの指定を行わなくても整数を代入すれば、変数は整数型にされるが、この指定により実行時間が短縮され、メモリが節約される。
 - 4 変数に%を付加せずに整数型に定義する時使う。
 - 5 -32768~+32767まで入力、表示可能。

プログラム例: Sで始まる変数を整数に宣言し、SEIに値を代入して出力する。

```
10 DEFINT S
20 SEI=3.141592654#
30 PRINT "SEI=";SEI
40 END
```

実行例

SEI= 3

参照: DEFSNG, DEFDBL, DEFSTR

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。ただし、カタカナ(アーン)、英字の小文字についても行える。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文に加えて、DEFINT @ は、カナ文字で始まる変数に対する指定。

SMC ・指定のない変数は単精度とみなされるので、特徴3はない。

MZ35 ・PC3と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

暗黙の単精度指定 DEFSNG

デファイン・シングル

形式: DEFSNG A-C, S-T

省略可 *省略・繰り返し可

機能: 変数の型を単精度に定義する。

- 特徴: 1 指定文字で始まる変数を単精度に型指定する。指定文字は最初の1文字のみ。
 2 変数毎の型宣言文字が、常にこの指定より優先する。
 3 特にこの指定を行わなくても変数はすべて単精度に初期設定されているが、この指定により実行時間が短縮される。
 4 7桁まで入力、6桁まで表示可能(7桁目は四捨五入される)。
 5 変数に!を付加せずに単精度に定義する時使うと便利である。

プログラム例: Tで始まる変数を単精度数に宣言し、TSEに値を代入して出力する。

```
10 DEFSNG T
20 TSE=3.141592654#
30 PRINT "TSE=" ; TSE
40 END
```

実行例

TSE= 3.14159

参照: DEFINT, DEFDBL, DEFSTR

APL —

TRS₁ —

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 DEFREAL アーオ
 ・本文と同じ。ただし、カナ(アーン)、英字の小文字についても行える。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。また、DEFSNG @ は、カナ文字で始まる変数に対する指定。これに関連して、暗黙の倍々精度指定には、DEFINを用いる。

SMC DEF FLT A-C, S-T
 ・機能は本文と同じ。

MZ35 ・PC3と同じ。ただし、12桁まで有効。

X1Hu ・本文と同じ。ただし、特徴4は単精度型による。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

暗黙の倍精度指定 DEFDBL

デファイン・ダブル

形式：DEFDBL B-C, S-T

省略可 *省略・繰り返し可

機能：変数の型を倍精度型に定義する。

- 特徴：1 指定文字で始まる変数を、倍精度数に指定する。指定文字は最初の一文字のみ。
 2 変数毎の型宣言文字が常にこの指定より優先する。
 3 17桁まで入力、16桁まで出力可能（17桁目は四捨五入される）。
 4 変数に#を付加せずに倍精度数に定義する時使う。

プログラム例：Dで始まる変数を倍精度数に宣言し、DSEに値を代入して出力する。

```
10 DEFDBL D
20 DSE=3.141592654#
30 PRINT"DSE=";DSE
40 END
```

実行例

DSE= 3.141592654

参照：DEFINT, DEFSNG, DEFSTR

APL —

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。また、

DEFDBL @ は、カナ文字で始まる変数に対する指定。

SMC —

MZ35 —

X1Hu ・本文と同じ。ただし、特徴3は倍精度型による。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

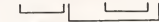
PC100 ・精度と表示は、ともに14桁である。

MSX ・本文と同じ。

暗黙の文字型指定 DEFSTR

デファイン・ストリング

形式：DEFSTR S-T, W-Z



省略可 *省略・繰り返し可

機能：変数の型を文字型に定義する。

- 特徴：1 指定文字で始まる変数を文字型に指定する。指定文字は最初の1文字のみ。
 2 変数毎の型宣言文字が常にこの指定より優先する。
 3 変数に\$を付加せずに文字型に定義する時使う。

プログラム例：Sで始まる変数を文字型に型宣言し、S1とS2にそれぞれ文字列を代入して、それらを加えたものをS3に代入し、出力する。

```
10 DEFSTR S
20 S1="ABCDEFGF"
30 S2="HIJKLMN"
40 S3=S1+S2
50 PRINT S3
60 END
```

実行例

ABCDEFGHIJKLMN

参照：DEFINT, DEFSNG, DEFDBL

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 ・本文と同じ。ただし、カナ(ア～ン)、英文字の小文字についても行える。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。また、DEFSTR @ は、カナ文字で始まる変数に対する指定。

SMC ・本文と同じ。

MZ35 ・PC3と同じ。なお、DEFSTR / とすると@型文字変数となる。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

表示文字の字形定義 DEFCHR\$

デファイン・キャラクタ\$

形式：DEFCHR\$ (224) = E\$

内部コード 字形

機能：表示する文字の字形を定義する。

- 特徴：1 内部コード224 (&HE0) から255 (&HFF) の文字の字形を定義する。
2 字形は文字式で与えられ、その16進数により字形ビット・パターンが定義される。

プログラム例：コード253の字形(人)を(虫)に定義する。

```
10 A$="007F497F00605007"
20 DEFCHR$(253)=A$
30 PRINT "コト" 1-9 の人 が けいけい。"
40 END
```

実行例

```
RUN
コト 1-9 の虫 が けいけい。
```

```
LIST
10 A$="007F497F00605007"
20 DEFCHR$(253)=A$
30 PRINT "コト" 1-9 の人 が けいけい。"
40 END
```

参照：

APL ・該当なし、ただしシェイプ・テーブルを用いれば同様のことが可。

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・引数の範囲は128～159と224～225。

レベル₃ —

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 ・本文参照。

SMC DEFONT 224, E\$
・内部コードは0から255までの範囲で指定する。

MZ35 —

X1Hu DEFCHR\$ (10) = A\$ + B\$ + C\$
・256文字まで定義できる。
・フォントはRGBの3色それぞれのパターンが必要。
・ユーザ定義とシステムのキャラクタの混在はできない。
CGPAT\$ (&H40)
・キャラクタ・コードに対応した32バイトの文字列によるパターンを与える。
CGEN 0
・キャラクタの切り替えを行う。
0：ROM, 1：ユーザ定義。

MB16 —

IBM55 —

PC100 —

MSX —

ベース・アドレスの設定 DEF SEG

デファイン・セグメント

形式: DEF SEG=&H0000

機能: セグメントのベース・アドレスの設定。

特徴: 数値は &H0 ~ &HFFFF の範囲で設定するが、実際のアドレスはこの値を4ビット左シフトした値を指す。

プログラム例: ベース・アドレスを変更して違う値の POKE 文が実は同じアドレスを指すことを示している。

```
100 DEF SEG=&H40
110 FOR I=0 TO 15
120 POKE &H100+I,I
130 NEXT I
140 DEF SEG=&H50
150 FOR J=0 TO 15
160 PRINT HEX$(PEEK(J)): " ";
170 NEXT J
180 END
```

実行例

0 1 2 3 4 5 6 7 8 9 A B C D E F

参照: BLOAD, BSAVE, CLEAR

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 —

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT — 本文と同じ。

HC —

FP11 —

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 — 本文と同じ。

PC100 — 本文参照。

MSX —

配列の定義 DIM

ディメンジョン

形式: DIM B (100), F\$ (150)

*省略・繰り返し可

機能: 配列変数の添字の最大値を指定し、その変数にメモリ領域を割り当てる。

- 特徴: 1 DIM 文で宣言しない配列を用いたときは、添字の最大は10。
 2 DIM 文で宣言した配列のすべての要素の値は、数値配列は0、文字配列は空に設定される。
 3 添字の指定は、数値または算術式で行う。
 4 指定した添字の最大値よりも大きい添字が用いられたときは、エラーが発生する。

プログラム例: 配列を用いて、乗算の表を作る。

```
10 DIM B(2,2)
20 FOR I=0 TO 2:FOR J=0 TO 2
30 B(I,J)=I*J
40 NEXT J:NEXT I
50 PRINT " * ";
60 FOR I=0 TO 2:PRINT I;:NEXT I
70 PRINT
80 FOR I=0 TO 2:PRINT I;
90 FOR J=0 TO 2:PRINT B(I,J);:NEXT J
100 PRINT:NEXT I
110 END
```

実行例

```
* 0 1 2
0 0 0 0
1 0 1 2
2 0 2 4
```

参照: OPTION BASE, ERASE, VTCLEAR, 配列

APL ・本文と同じ。ただし、次元は1～88の範囲。
 ・文字列配列の文字列長の最大値は255。

TRS1 ・本文と同じ。

CPM ・本文と同じ。ただし、添字の最小値はOPTION BASE 文で、0か1に設定できる。

M223 ・本文と同じ。ただし、次元は2次元まで。
 ・1次元のとき、数値配列の添字は0～65000の範囲。
 ・文字列変数の文字列長は通常10であるが、DIM F\$ (20)により、文字列長(例では20文字)を1～255の範囲で指定できる。
 ・文字列の2次元配列は、DIM F\$ (150, 20)で行と最大文字列長を指定する。

MZK ・本文と同じ。ただし、添字の最大値は255。

CBM ・本文と同じ。ただし、次元は1～255の範囲。添字の最大値は32767。

PC8 ・本文参照。

TRS11 ・本文と同じ。

PC3 ・本文と同じ。ただし、文字列変数の文字列長は、通常16であるが、DIM F\$ (150) * 20により、文字列長を指定できる。

IF82 ・CPMと同じ。

レベル3 ・本文と同じ。

C180 ・CPMと同じ。ただし、次元は2次元まで、添字の最大値

は65535。
 ・文字列変数の文字列長は、通常18であるが、DIM F\$ (150) * 20により文字列長を1～508の範囲で指定できる。

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・CPMと同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。ただし、次元は1～15の範囲。
 ・文字列変数の文字列長は、通常32であるが、DIM F\$ * 20 (100)により、文字列長を1～255の範囲で指定できる。

PC88 ・CPMと同じ。

N52 ・次元は3次元までとする。
 添字の範囲は0～32767。

PC6 ・次元は3次元まで。

MLT ・CPMと同じ。

HC ・本文と同じ。

FP11 ・CPMと同じ。

SMC ・CPMと同じ。

MZ35 ・PC3と同じ。ただし、特徴1はない。

X1Hu ・本文と同じ。ただし、次元は255次元まで。

MB16 ・CPMと同じ。

IBM55 ・CPMと同じ。

配列下限値の指定 OPTION BASE

オプション・ベース

形式：OPTION BASE 1

↑ ↑
必要 必要

機能：配列の下限値を 0 または 1 のどちらかに決める。

- 特徴：**
- 1 定数に 0 を用いると、OPTION 文以後宣言された配列の下限は 0 になり、1 を用いると下限は 1 となる。
 - 2 OPTION 文は一つのプログラム中、一度だけ使用可能（配列宣言の前に書く）。
 - 3 OPTION 文を使わなかった場合は、暗黙に 0 に指定される。
 - 4 配列の添字と要素数を一致させるとき、OPTION BASE 1 を使うと便利である。

プログラム例：配列 B の下限値を 0 にする。

```
10 OPTION BASE 0
20 DIM B(4)
30 FOR I=0 TO 4
40 B(I)=1
50 PRINT I,B(I)
60 NEXT I
70 END
```

実行例

0	1
1	1
2	1
3	1
4	1

参照：DIM, 配列

APL —

TRS₁ —

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・本文と同じ。

レベル₃ —

C180 ・本文参照。
・下限値および上限値は LDIM,
UDIM 関数でわかる。

M243 —

MZB —

BUB ・本文と同じ。

FM8 —

PSP —

PC88 ・本文と同じ。

N52 —

PC6 —

MLT ・C180 と同じ。

HC ・本文と同じ。

FP11 ・特徴 2 はない。

SMC ・本文と同じ。

MZ35 —

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX —

メモリ・クリア CLEAR

クリア

形式: CLEAR 500, 50000

省略可

機能: すべての数値変数を 0 に, 文字変数をナル・ストリングにリセットする。

- 特徴: 1 初めの引数で, 指定されたバイト数だけ文字領域を確保する。省略すると 300 に設定される。
- 2 第 2 の引数で, BASIC で使えるメモリの上限のアドレス¹⁾を指定する。そして, それ以降に, 10 進アドレス 59903 (16 進で E9FF) までの機械語ユーザ・サブルーチンを格納するときに使う。省略すると最上位アドレス 59903 に設定される。
- 3 第 2 の引数を指定すると, 初めの引数を省略できない。

用語: ¹⁾アドレス 記憶場所などを一意的に識別する番号。番地ともいう。

プログラム例: 変数の初期化をして印字する。

```

10 DIM B(7)
20 A=123
30 FOR I=0 TO 7:B(I)=I:NEXT I
40 C#=3.14159265358979#
50 E$="BASIC"
60 PRINT "A=";A;"C#=";C#
70 PRINT "B(I):";
80 FOR I=0 TO 7:PRINT B(I);:NEXT I
90 PRINT:LPRINT"E$=";E$
100 CLEAR
110 PRINT "    *** clear ***"
120 PRINT "A=";A;"C#=";C#
130 PRINT "B(I):";
140 FOR I=0 TO 7:PRINT B(I);:NEXT I
150 PRINT:PRINT "E$=";E$

```

実行例

```

A= 123          C#= 3.14159265358979
B(I): 0  1  2  3  4  5  6  7
E$=BASIC
    *** clear ***
A= 0            C#= 0
B(I): 0  0  0  0  0  0  0  0
E$=

```

参照: DEFUSR, VCLEAR

APL CLEAR および LOMEM
・CLEAR 文は本文と同じ。ただし、引数は使えない。また LOMEM 文で BASIC で使用するメモリの上限を指定する。

TRS: CLEAR500

・引数の一つとり, それで文字領域を確保する。引数を省略すると, 50 バイトに設定される。

CPM CLEAR, 49152, 500

・初めの引数で, BASIC で使用するメモリの上限を, 次の引数で BASIC のためのスタック領域を設定する。ただし, これらを指定するときは CLEAR の後にコンマ (,) が必要。文字領域は自動的に確保される。

M223 ・CLEAR は画面消去用。

・同命令は VCLEAR。ただし、引数はもたない。

・VTCLEAR は変数領域をも消してしまう。

MZK CLEAR および LIMIT

・APL と同じ。ただし、LIMIT 文の引数は変数も使える。また LIMIT MAX で BASIC 領域の最大限が指定される。

CBM ・APL と同じ。

PC8 ・本文参照。

TRS_{II} ・TRS_I と同じ。

PC3 —

IF8₂ ・CPM と同じ。ただし、メモリの上限は 65535 で、スタック領域の指定を省略すると 256 に設定される。

レベル₃ ・引数を省略すると, 文字領域を 300 バイト, メモリの上限は 32767 (&H7FFF) に設定される。

C180 —

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・CPM と同じ。ただし、CLEAR の後にコンマ (,) はいらない。

FM8 ・アドレスの上限は 28671。

PSP ・APL と同じ。

PC88 CLEAR 300, &HF000, 1000

・CPM と同じ。ただし、第 1 の引数で文字領域の大きさを指定できるが、意味はもたない。

N52 CLEAR FILE, 5, 500

・機能は本文のほかに、上記の方法でオープンして使用するファイルの個数、および文字領域の大きさを設定することができる。

・最初の引数はファイルの個数で、1 から 15 まで指定でき、初期値は 3 である。第 2 の引数は特徴 1 と同じ (共に省略可)。

PC6 ・本文と同じ。

MLT ・CPM と同じ。

HC CLEAR 200, 256

・初めの引数を省略すると 200 に設定される。

・第 2 の引数は RAM ファイルの大きさを指定する。

MEMSET &H0D00

メモリ・クリア CLEAR

クリア

<p>・機械語プログラムは、BASIC プログラムのテキスト・エリアの前におかれるので、BASIC で使用するメモリの下限を設定する。</p>	<p>めの引数を省略すると 200 バイトに設定される。</p>		
<p>FP11 ・引数を省略すると、現在の値が指定される。電源 ON 時には、それぞれ 1023、65534 (&HFFFFE) に設定されている。</p>			
<p>SMC CLEAR ・引数はもたない。 ・機能は本文と同じ。 MCLEAR M1, M2 ・BASIC 領域のロー・メモリを M1 で、ハイ・メモリを M2 で指定する。 ・引数の省略は、初期状態にリセットされる。</p>			
<p>MZ35 ERASE* &A000</p>			
<p>X1Hu CLEAR ・変数クリア (CLR も同義) CLEAR &HC000 LIMIT &HC000 ・BASIC で使用する上限を指定する。 ・変数はクリアされず、スタックのみクリアされる。</p>			
<p>MB16 ・CPM と同じ。ただし、スタック領域の省略時には 512 バイトまたは、使用可能なメモリ領域の 1/8 のいずれか小さいほう。</p>			
<p>IBM55 ・CPM と同じ。ただし、スタック領域の指定を省略すると 512 または、使用可能な記憶機構の 1/8 のどちらか小さいほうが、設定される。</p>			
<p>PC100 ・PC88 と同じ。</p>			
<p>MSX ・本文と同じ。ただし、初</p>			

変数の削除 ERASE

イレース

形式：ERASE B, F\$

*省略・繰り返し可

機能：プログラムから、指定した配列変数を取り除く。

- 特徴：
- 1 指定した配列変数を消去する。
 - 2 削除した配列と同一変数名の配列変数を、再び DIM 文で宣言できる。
 - 3 削除した配列変数に割り当てられていたメモリ領域を、ほかの目的に使用できる。
 - 4 配列の添字の最大値の変更など、配列を再定義するのに便利である。

プログラム例：配列を再定義する。

```

10 DIM B(7)
20 FOR I=0 TO 7:B(I)=I:PRINT B(I);:NEXT I
30 PRINT
40 ERASE B
50 DIM B(12)
60 FOR I=5 TO 12:B(I)=I:NEXT I
70 FOR I=0 TO 12:PRINT B(I);:NEXT I
80 END

```

実行例

```

0 1 2 3 4 5 6 7
0 0 0 0 0 5 6 7 8 9 10 11 12

```

参照：DIM, VTCLEAR

APL —

TRS₁ —

CPM ・本文と同じ。

M223 KILL B, F\$

- ・機能は本文と同じ。さらに、変数も消去できる。
- ・VTCLEARで、プログラムからすべての変数領域を取り除くことができる。

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。レベル₃ —

C180 —

M243 ・M223と同じ。

MZB —

BUB ・本文と同じ。

FM8 —

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC —

MZ35 ERASE B

X1Hu —

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

関数定義文 DEFFN

デファイン・ファンクション

形式：DEF FNA (A, B) = SIN (A) * B

関数の名前 *省略・繰り返し可
省略可
パラメータ・リスト

機能：新しくユーザが関数を定義し、名前をつける。

- 特徴：1 関数の定義式に使われた変数名は、パラメータ・リスト中に必ずしも必要はない。もしあれば、関数が呼ばれた時にパラメータの値が与えられ、ない場合には、その変数の現在の値が使われる。
2 数値関数でも、文字関数でもよい。
3 関数名は、FNに続く英字で始まる任意個の文字列。ただし、FN以後の2文字のみが識別に有効。引用のときは、FNを冠した関数名を使う。
4 頻繁に用いる数式を定義しておくくと便利である。

プログラム例：関数 FNA (A, B) = A*B+C を定義し用いる。

```
10 DEFFNA (A, B) = A*B+C
20 A=3: B=4: C=5
30 D=FNA (A, B)
40 E=FNA (A, C)
50 LPRINT "D=", D
60 LPRINT "E=", E
70 END
```

実行例

```
D=          17
E=          23
```

参照：FNEND

APL DEF FN A(I) = SIN(I)
・パラメータは一つ。

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 ・FN END 文を用いて複数行のプログラムを一つの関数として定義できる。関数名には変数名と同様の制限がある。

MZK ・APL と同じ。

CBM ・MZK と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・関数名は、変数名と同様の長い文字列が許される。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・FN に続く文字は A～Z のうちの一文字。

PC88 ・名前の文字数は40字までゆるされる。

N52 ・本文と同じ。

PC6 DEF A = SIN (I)
・パラメータは一つ。

MLT ・C180 と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・引数は10個までで、変数名は255文字までである。
・再帰的定義も可能。
・関数定義に際して、COND, PROG, SETQ, EVAL関数を用いることにより、複数行のプログラムを一つの関数にすることが可能。

MZ35 ・本文と同じ。ただし、特徴1は不可。パラメータ・リストに与えられていない変数は0とみなされる。
・特徴3では8文字まで有効。

X1Hu ・C180 と同じ。

MB16 ・PC88 と同じ。

IBM55 ・PC88 と同じ。

PC100 ・PC88 と同じ。

MSX ・本文と同じ。

変数引き渡し宣言 COMMON

コモン

形式：COMMON A, B

*省略、繰り返し可

機能：CHAIN 文が実行されたときに、指定した変数の内容を引き渡す。

- 特徴：
- 1 必ず CHAIN 文とペアで用いられなければならない。
 - 2 非実行文なので、プログラムのどこにあっておかまわれないが、少なくとも CHAIN 文の前には必要である。
 - 3 同じ変数を重複して宣言してはならない。
 - 4 配列変数は A () のように " () " を付けて表現する。
 - 5 すべての変数を引き渡す場合には、CHAIN 文の ALL オプションを用いるほうが便利である。

プログラム例：最初のプログラムで計算した B (1 から 10), A1, A2 の内容を、次のプログラムに引き渡し、プリントさせる。

```

10 'Program for COMMON
20 COMMON B(),A1,A2
30 DIM B(15):A1=0:A2=0
35 LPRINT 'B=';
40 FOR I=1 TO 10
50 B(I)=I*I
60 A1=A1+B(I)
70 A2=A2+A1*B(I)
80 LPRINT B(I);:NEXT I
90 LPRINT 'A1=';A1,'A2=';A2
100 CHAIN 'procom'
110 END

```

```

10 'test program for common
20 LPRINT 'after chain':LPRINT 'B=';
30 FOR I=1 TO 10
40 LPRINT B(I);:NEXT I
50 LPRINT 'A1=';A1,'A2=';A2
60 END

```

実行例

```

B= 1 4 9 16 25 36 49 64 81 100 A1= 385 A2= 86779
after chain
B= 1 4 9 16 25 36 49 64 81 100 A1= 385 A2= 86779

```

参照：CHAIN

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・本文と同じ。レベル₃ —

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 —

PSP ・本文と同じ。

PC88 ・本文参照。

N52 —

PC6 —

MLT ・本文と同じ。

HC —

FPI1 —

SMC —

MZ35 —

X1Hu —

MB16 ・本文と同じ。

IBM55 ・本文と同じ。ただし、特徴 2 はない。

PC100 ・本文と同じ。

MSX —

機械語開始アドレス指定 DEFUSR

デファイン・ユーザ

形式：DEFUSRO=&HE000

□
省略可

機能：機械語ユーザ・サブルーチンの開始アドレスを指定する。

- 特徴：1 等号(=)の左辺は、0～9までの任意の数字である。右辺でユーザ・サブルーチンの開始アドレスを指定する。省略すると、0が設定される。
- 2 開始アドレスは、10進、16進の双方とも使える。
- 3 ユーザ・ルーチンの開始番地を指定し直すために、一つのプログラムにいくつもDEFUSR文をおける。
- 4 機械語ルーチンの番号を指定して、BASIC プログラム中で使うとき用いる。

プログラム例：指定数だけ(*)を画面表示する。

```
10 DEF USR1=&HE000
20 FOR I=&HE000 TO &HE00C
30 READ A$
40 POKE I,VAL("&H"+A$)
50 NEXT I
60 X=USR1(7)
70 PRINT
80 X=USR1(11)
90 END
100 DATA eb,1a,47,3e,2a,cd,57,
    02,05,c2,05,e0,c9
```

実行例

```
*****
*****
```

参照：USR

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 —

MZK ・該当なし。USR文で直接機械語実行開始番地を指定する。引数の引き渡しはできない。

CBM —

PC8 ・本文参照。

TRS₁₁ ・アドレスは-32768から32768まで(10進)をとる。

PC3 ・該当なし。ただし、USR文で引数、機械語実行開始番地の双方を指定できる。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 —

M243 —

MZB ・MZKと同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 —

PC6 —

MLT DEF USR [*2] = オフ
セット

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC DEF ANNO = &HE000
・機能は本文と同じ。

MZ35 USR 参照。

X1Hu ・本文と同じ。

MB16 ・実行開始番地は、直前の“DEF SEG”文により定義されたセグメント値からのオフセット値として指定。
・“DEF”と“USR”の間はスペースを1個あけなければならない。

IBM55 ・アドレスは0から65535の範囲の整数式。
・その他はMB16と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

並列処理部の定義 PARACT, PAREND, URGENCY パラアクト, パラエンド, アージェンシー

APL —	SMC —		
TRS ₁ —	MZ35 —		
CPM —	X1Hu —		
M223 —	MB16 —		
MZK —	IBM55 —		
CBM —	PC100 —		
PC8 —	MSX —		
TRS ₁₁ —			
PC3 —			
IF8 ₂ —			
レベル ₃ —			
C180 ・ 本文参照.			
M243 —			
MZB —			
BUB —			
FM8 —			
PSP —			
PC88 —			
N52 —			
PC6 —			
MLT —			
HC —			
FP11 —			

繰返し FOR

形式: FOR I=1 TO 20 STEP 3

変数名 初期値 最終値 省略可
増分

機能: 指定した回数だけ一連の命令群を繰返して実行する。

- 特徴: 1 増分の値は負数でもよい。この時初期値は、最終値より大きくなければならない。
 2 STEP 部が省略された場合は、増分は1とみなされる。
 3 条件の真偽にかかわらず、最低1回は、ループは実行されるので注意が必要である。
 4 ネスティング(多重構造)¹⁾にすることができる。
 5 対応する NEXT 文が必要。
 6 初期値、最終値、増分は、定数および算術式である。
 7 一連の作業を何回も繰返す時、変数を一定の割合で増減させて用いる時などに便利である。

用語: ¹⁾ネスティング プログラム中で、ループの中にループがあったり、サブルーチン内でサブルーチンを呼んだりといった、プログラムが、多重構造になっていることをいう。

プログラム例: I を1から3ずつ加えて出力し、20を越えたら終了。

```
10 FOR I=1 TO 20 STEP 3
20 LPRINT I
30 NEXT I
40 END
```

実行例

```
1
4
7
10
13
16
19
```

参照: NEXT, UNTIL, WHILE

APL ・本文と同じ。

TRS_I ・本文と同じ。

CPM ・本文と同じ。ただし、特徴3は、最初から条件が満たされている場合、FOR~NEXT ループは1回も実行されず、NEXT 文の次の文にプログラムの制御が移る。

M223 ・CPMと同じ。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。

IF8₂ ・CPMと同じ。レベル₃ ・CPMと同じ。

C180 ・FOR 文と NEXT 文は1対1の対応。

M243 ・CPMと同じ。

MZB ・本文と同じ。

BUB ・CPMと同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・CPMと同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・特徴3は、初期値が終値より大きくかつSTEPの値が正のとき、あるいは、初期値が終値より小さくかつSTEPの値が負の場合、ループを飛び越して、NEXT 文の次のプログラムに制御が移る。

HC ・本文と同じ。

FP11 ・CPMと同じ。

SMC ・CPMと同じ。

MZ35 ・本文と同じ。

X1Hu ・CPMと同じ。

MB16 ・CPMと同じ。

IBM55 ・CPMと同じ。

PC100 ・CPMと同じ。

MSX ・本文と同じ。

繰り返しの端末 NEXT

ネクスト

形式：NEXT I, J
└┘

省略可 *省略・繰り返し可

FOR 文と対応する変数名

機能：FOR 文と対をなして、ループの終わりを示す。

- 特徴：1 多重ループが同じ場所で終わる場合は、すべてのループに対して、一つの NEXT 文ですませられ、その場合に変数は複数となり、内側のループの変数を左から書く。
 2 NEXT 文が一番近い FOR 文に対応する場合には、変数を省略できる。

プログラム例：I を 20 から三ずつ減じて出力し、1 より小さくなったら終了。

```
10 FOR I=20 TO 1 STEP -3
20 LPRINT I
30 NEXT I
40 END
```

実行例

20
 17
 14
 11
 8
 5
 2

参照：FOR

APL ・本文と同じ。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 NEXT I
 ・I は省略不可。

MZK NEXT I, J
 ・I は省略不可。、J は省略・繰り返し可。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 ・本文と同じ。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・M223 と同じ。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・M223 と同じ。

FM8 ・本文と同じ。

PSP ・M223 と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・本文と同じ。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

条件文 IF

形式: IF A2 >= 2 THEN GOTO 70

論理式

省略可

省略可

実行文

ELSE GOTO 170

省略可

省略可

実行文

機能: プログラムの実行の流れの制御を, 論理式の条件に従って行う。

- 特徴:
- 1 THEN, ELSE の後ろは実行文である。
 - 2 ELSE を省略した場合には, THEN を省略して GOTO で行番号を指定してもよい。
 - 3 ネスティング(多重構造)を行える。この場合, もし文中の ELSE 節と THEN 節の数異なる時には, それぞれの ELSE は最も近く, まだ対応のない THEN に対応する。
 - 4 THEN の後が複文の場合に, 条件が満たさなければ複文を実行せずに, プログラム制御は次の行へ移る。
 - 5 プログラム制御の流れを, 式の値でいくつかに分岐させる場合, その値にバラツキがあるときに便利である。

プログラム例: X, Y の値を入力し, その大小関係を判定する。

```
10 INPUT "X= ";X
20 INPUT "Y= ";Y
30 IF X<>Y THEN IF X>Y THEN PRINT X;">" ;Y
    ELSE PRINT X;"<" ;Y
40 IF X=Y THEN PRINT X;"=" ;Y
50 IF (X<>0) OR (X<>Y) THEN 10
60 PRINT "END"
70 END
```

実行例

```
X= ? 15.32
Y= ? 3.25
15.32 > 3.25
X= ? -35.2
Y= ? 0.36
-35.2 < 0.36
X= ? 25.03
Y= ? 25.03
25.03 = 25.03
X= ? 0
Y= ? 0
0 = 0
END
```

APL ・本文と同じ。ただし、ELSE以後は使えない。

TRS₁ ・本文と同じ。ただし、THENはほかの行番号へ分岐する場合を除き省略可。

CPM ・本文と同じ。ただし、THENを省略してGOTOを使用した場合でも、ELSEを使用できる。

M223 ・本文と同じ。ただし、THEN, ELSEは省略できない。

MZK ・APL と同じ。

CBM ・APL と同じ。

PC8 ・本文参照。

TRS_{II} ・TRS_I と同じ。

PC3 ・本文と同じ。さらに、行番号の代わりにラベル名、数式をも書ける。数式の値は小数点以下は切り捨てられる。

IF8₂ ・CPM と同じ。

レベル₃ ・本文と同じ。

C180 ・END IF文を用いて、THENのあとの複数行を1ブロックとすることができる。
・これに関連して、CASEをSELECT, END SELECTとともに用いることにより、多条件分岐が可。

M243 ・M223 と同じ。

MZB ・APL と同じ。

BUB ・本文と同じ。

FM8 ・CPM と同じ。

PSP ・THEN の代わりに、GOSUB 文, GOTO 文を使うことができる。

PC88 ・CPM と同じ。さらに、行番号の代わりにラベル名を使用できる。

N52 ・CPM と同じ。

PC6 ・APL と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・CPM と同じ。また、分岐先は、行番号のほかにプログラム・エリア番号を指定できる。

SMC ・THEN の代わりに、GOTO 文, GOSUB 文, LET 文 (LET は省略できない) が使える。

MZ35 ・PC3 と同じ。

X1Hu ・PC88 と同じ。

MB16 ・CPM と同じ。

IBM55 ・CPM と同じ。ただし、THENの前および、IF…GOTO…ELSEにおける、GOTO, ELSEの前に(,)をつけてもよい。
・ELSEのあとのGOTOは省略してはならない。

PC100 ・PC88 と同じ。

MSX ・CPM と同じ。

参照:

飛び越し GOTO

ゴーツー

形式：GOTO 70

行番号

機能：通常のプログラムの流れから、無条件に指定された行に制御を移す。

特徴：1 指定する行番号先は、非実行文（たとえば REM 文）でもよく、この場合はその行以後の最初の実行文から実行が続けられる。
2 GO TO も可。

プログラム例：“STOP”以外の文字列が入力されたらそれを表示する。“STOP”の場合行番号30へ飛び中断する。

```
10 INPUT E#
20 IF E#<>"stop" THEN PRINT"モシ`レツ=";E#
   :GOTO 10
30 STOP
40 END
```

実行例

```
? abc
モシ`レツ=abc
? タテ
モシ`レツ=タテ
? ##$%
モシ`レツ=##$%
? stop
Break in 30
Ok
cont
Ok
```

参照：ON GOTO, IF

APL ・本文と同じ。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 GOTO "LABEL"
GOTO (A1+2)
・行番号の代わりにラベル名と数式を書ける。数式の値は小数点以下切り捨てられる。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・本文と同じ。

M243 ・本文と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・行番号の代わりにラベル名も書ける。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 GOTO 70
GOTO PROG1
・行番号の代わりにプログラム・エリア番号も指定できる。この場合、指定されたエリアの先頭行へ分岐する。
・GO と TO の間のスペースは一つのみ許される。

SMC ・PC88 と同じ。

MZ35 ・PC3 と同じ。

X1Hu ・PC88 と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・PC88 と同じ。

MSX ・本文と同じ。

多方向への分岐 ON GOTO

オン・ゴートウ

形式: ON A1+3 GOTO 70, 170, 270

算術式

省略, 繰り返し可

機能: 式の値を計算し, その結果によって与えられた幾つかの行番号の文へプログラムの制御を移す。

- 特徴:
- 1 式の値の小数部分は切り捨てられる。
 - 2 式の値が負の場合はエラーとなる。
 - 3 式の値が0または行番号の個数より大きい場合は, プログラムの制御は次の行へ移り, エラーは起こらない。
 - 4 プログラム制御の流れを式の値によって, 幾つかに分岐する場合, 式の値にばらつきが少ない場合に用いると便利である。

プログラム例: Iの値を3, 4と変化させI-2の値によって, 行番号40, 70へそれぞれ分岐する。

```

10 I=3
20 ON I-2 GOTO 40,70
30 END
40 LPRINT "LINE NO. 40 I-2=";I-2
50 I=4
60 GOTO 20
70 LPRINT "LINE NO. 70 I-2=";I-2
80 END

```

実行例

```

LINE NO. 40 I-2= 1
LINE NO. 70 I-2= 2

```

参照: ON GOSUB

APL ・ 本文と同じ。

TRS_I ・ 本文と同じ。

CPM ・ 本文と同じ。ただし, 式の値の小数部分は四捨五入される。式の値が0または行番号の個数より大きい場合でもエラーが起こり, メッセージが表示される。

M223 ・ 本文と同じ。ただし, 式の値がマイナスでもエラーが起こらず, 次の行にプログラムの制御が移る。

MZK ・ M223 と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 本文と同じ。さらに, 行番号の代わりにラベル名も使用できる。

IF8₂ ・ CPM と同じ。

レベル₃ ・ 式の値の小数部分は四捨五入される。

C180 ON A1+3 GOTO 70, 170
ELSE 実行文
・ 式の値の小数部分は四捨五入される。式の値が0以下または行番号の個数より大きい場合は, ELSEに続く実行文を実行する。ELSEがない場合にはエラー表示"E108"がある。

M243 ・ M223 と同じ。

MZB ・ M223 と同じ。

BUB ・ CPM と同じ。

FM8 ・ M223 と同じ。

PSP ・ M223 と同じ。

PC88 ・ PC3 と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 小数部分は四捨五入される。また, 特徴2では, 負の場合と255より大きい場合にエラーとなる。

HC ・ 算術式の値は0から255までである。

FP11 ・ M223 と同じ。また, 分岐先は, 行番号のほかにプログラム・エリア番号を指定できる。

SMC ・ PC3 と同じ。ただし, 式の値の小数部分は四捨五入され, 式の値が256以上のときもエラーとなる。

MZ35 ・ PC3 と同じ。なお, 特徴2では M223 と同じ。

X1Hu ・ レベル₃ と同じ。
・ 行番号の代わりにラベル名も使える。
・ ほかに ON~RETURN
ON~RESTORE
ON~RESUME
文がある。

MB16 ・ レベル₃ と同じ。

IBM55 ・ MLT と同じ。

PC100 ・ 式の値の小数部分は四捨

多方向への分岐 ON GOTO

オン・ゴートゥー

<p>五入される。 ・ラベル名も使える。</p>			
<p>MSX ・本文と同じ。</p>			

ループ条件文 WHILE

形式：WHILE A1<A2

機能：条件式が真である間、一連の命令を繰り返し実行する。

- 特徴：1 特定の命令と共にマルチ文において用いられ、条件式が真である間、そのマルチ文を繰り返す。
 2 WHILE を含むマルチ文の中の GOSUB, DEF 文の使用はできない。
 3 最後に $A1 \geq A2$ となったときは実行をしないので注意。よって UNTIL $A1 > A2$ とは異なる。
 4 FOR~NEXT 文に WHILE を用いると、その機能が向上して便利である。

プログラム例：フィボナッチ数列の要素の値が30以上になるまで求める。

```

10      OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20      LET J = 1
30      LET K = 1
40      PRINT #1, K
50      :   PRINT #1, K
        LET I = J
        :   LET J = K
        :   LET K = I+J
        :   PRINT #1, K WHILE K < 30
60      CLOSE 1
70      END

```

実行例

```

1
1
2
3
5
8
13
21
34

```

参照：UNTIL, FOR, NEXT

APL —

TRS_I —

CPM 10 WHILE A1<A2

:

50 WEND

- ・条件式の値が真のときは WHILE とそれに対応する WEND の間の部分を繰り返し実行する。
- ・条件式の値が最初から真でないときは、WHILE と WEND の間は 1 回も実行されない。
- ・WHILE~WEND は多重化できる。

M223 ・本文参照。

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・CPM と同じ。レベル₃ —

C180 —

M243 ・M223 と同じ。

MZB —

BUB ・CPM と同じ。

FM8 ・CPM と同じ。

PSP —

PC88 ・CPM と同じ。

N52 —

PC6 —

MLT ・CPM と同じ。

HC ・CPM と同じ。

FP11 ・CPM と同じ。

SMC ・CPM と同じ。

MZ35 —

X1Hu ・CPM と同じ。なお、GOTO でループの外には出られない。

MB16 ・CPM と同じ。

IBM55 ・CPM と同じ。

PC100 ・CPM と同じ。

MSX —

ループ条件文 UNTIL

アンティル

形式：UNTIL A1>A2

機能：条件式が真になるまで、一連の命令を繰り返し実行する。

- 特徴：
- 1 特定の命令と共にマルチ文において用いられ、条件式が真になるまで、そのマルチ文を繰り返す。
 - 2 UNTILを含むマルチ文の中での GOSUB、DEF 文の使用はできない。
 - 3 条件式が真になった最後の 1 回はマルチ文を実行するので注意。よって WHILE A1<=A2 とは同じでない。
 - 4 FOR～NEXT 文に UNTIL を用いると、その機能が向上して便利である。

プログラム例：フィボナッチ数列の要素が30以上になるまで求める。

```

10      OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20      LET J = 1
30      LET K = 1
40      PRINT #1, K
      :   PRINT #1, K
50      LET I = J
      :   LET J = K
      :   LET K = I+J
      :   PRINT #1, K UNTIL K > 30
60      CLOSE 1
70      END

```

実行例

```

1
1
2
3
5
8
13
21
34

```

参照：WHILE, FOR, NEXT

APL —

TRS₁ —

CPM —

M223 ・ 本文参照。

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 ・ 本文と同じ。

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu 10 REPEAT

⋮
50 UNTIL A<1

- ・ 論理式が偽の間ループする。
- ・ GOTO でループの外には出られない。
- ・ WHILE～WEND と異なり、必ず REPEAT～UNTIL 間の文は、最低 1 回実行される。

MB16 —

IBM55 —

PC100 —

MSX —

ファイル終了の分岐 AT EOF #n

アット・エンド・オブ・ファイル

形式：AT EOF #5:100

↑ ↑ └──
 必要 必要 行番号

機能： エンド・オブ・ファイル (EOF) を検出したときに、指定した行番号へ飛ぶ。

- 特徴：**
- 1 CLOSE# を実行すると、この文は無効となる。
 - 2 TRAP 文よりも優先される。
 - 3 データの数があらかじめ不明なファイルから連続してデータを入力するとき、データの終わりを知るのに便利である。

プログラム例： AFILE を作り、データを書き込み、そのデータを入力してプリントする (実行例は RESTORE を参照)。

```

100 DIM B(1)
110 CREATE "AFILE/1",CONTINUOUS(10,9)
120 OPEN #1:"AFILE/1".INOUT,FIXED(10)
130 AT EOF #1:240
140 FOR I=1 TO 10
150 B(1)=I*3.14159
160 PUT #1:B
170 NEXT I
180 RESTORE #1
190 N=1
200 GET #1:B
210 PRINT "ジョックイ=";N;"インシュウ=";B(1)
220 N=N+1
230 GOTO 200
240 PRINT "END"
250 END

```

参照：EOF, ON ERROR GOTO

APL ・該当なし。ただし、
10 ONERR GOTO 170

⋮
行番号(エラー
処理部)

170 IF PEEK(222)=5 THEN 100
で同様のことが可。PEEK(222)で
エラー・コードを調べることがで
きる。エラー・コード5は EOF
の検出である (ERR 参照)。

TRS₁ ・該当なし。ただし、
IF EOF(5) THEN 100 をファ
イル入力の前に入れば同様のこ
とが可。5はファイル番号。

CPM ・TRS₁ と同じ。

M223 ・該当なし。ただし、
10 ON ERROR GOTO 170
⋮
170 IF ERR=11 THEN GOTO
100 で同様のことが可。ERRが11
であれば、ファイルの終了を示す。

MZK ・該当なし。ただし、
IF EOF(#5) THEN 100 をファ
イル入力の前に入れば同様のこ
とが可。

CBM —

PC8 ・TRS₁ と同じ。TRS₁₁ ・TRS₁ と同じ。

PC3 ・該当なし。ただし、
10 ON ERROR GOTO 170
⋮
170 IF ERN=209 THEN GOTO
100 で同様のことが可。ERNが209
であれば、ファイルの終了を示す。

IF8₂ ・TRS₁ と同じ。レベル₃ ・TRS₁ と同じ。

C180 ・本文参照。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・TRS₁ と同じ。FM8 ・TRS₁ と同じ。

PSP ・該当なし。ただし、
INPUT #5;E\$ EOF 100 で
同様のことが可。

PC88 ・TRS₁ と同じ。N52 ・TRS₁ と同じ。PC6 ・TRS₁ と同じ。MLT ・TRS₁ と同じ。

HC ・該当なし。ただし、IF
EOF(3) THEN CLOSE#1
ELSE GOTO 100 で同様のこ
とが可。

FPI1 ・TRS₁ と同じ。SMC ・TRS₁ と同じ。

MZ35 ・PC3 と同じ。

XIH_u ・TRS₁ と同じ。MB16 ・TRS₁ と同じ。IBM55 ・TRS₁ と同じ。PC100 ・TRS₁ と同じ。

MSX —

エラー処理分岐 ON ERROR GOTO

オン・エラー・ゴーツー

形式：ON ERROR GOTO 70

機能：エラーが発生したとき、プログラムの実行を指定したエラー処理ルーチンに移す。

- 特徴：1 プログラムの実行中にエラーが発生したとき、ON ERROR GOTO 文によって実行は中断されず、行番号で指定した行から始まるエラー処理ルーチンを実行し、エラーの発生による実行の停止を防ぐことができる。
- 2 ON ERROR GOTO 0 を実行すると、1 の特徴を禁止することができる。
- 3 エラー処理ルーチン内で、ON ERROR GOTO 0 を実行すると、エラー処理を行わずに実行を停止する。
- 4 指定した行番号の行が存在しないときは、エラーが発生する。
- 5 エラーが発生した時、1 回だけ飛ぶ。飛んだ後は ERR が 0 でなくても新しいエラーが発生しないかぎり飛ばない。
- 6 処理可能なエラーによる実行の停止を防ぐことができ便利である。

〈参考〉エラー処理ルーチンの実行中にエラーが発生したとき、実行は停止する。

プログラム例：逆数を求める。(0 による除算はエラーとなる)。

```
10 ON ERROR GOTO 100
20 INPUT A
30 PRINT "1/" ; A ; "=" ; 1/A
40 GOTO 20
100 IF ERR=11 THEN PRINT "ERROR"
110 RESUME 20
```

実行例

```
? 4
1/4 = .25
? 0
1/0 = ERROR
? -8
1/-8 = -.125
?
Break in 20
```

APL ONERR GOTO 70

・機能は本文と同じ。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・本文と同じ。

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。ただし、エラー処理部を行番号以外に、ライン・ラベル名、または数式の値(整数部のみ有効)の行番号で指定できる。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・処理可能なエラーが発生したときは、システムがエラー処理部へ自動的に実行を移す。

M243 ・本文と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・行番号の代わりにラベル名も使える。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・行番号にラベル名も使用できる。
・128 以上の番号のエラーが発生した時は、無効である。

MZ35 ・PC3 と同じ。

X1Hu ・PC88 と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・PC88 と同じ。

MSX ・本文と同じ。

参照：RESUME, ERR, ERL, ERROR, CAUSE

エラーの模擬発生 ERROR

形式: ERROR A1+3

機能: 算術式の値と同じエラーの発生を模擬するか、ユーザによるエラー番号の定義を可能にする。

- 特徴: 1 算術式は整数でなければならない。ただし、範囲は 0~255 まで。
- 2 システムでエラー番号として使用されている値と算術式の値が一致すると、そのエラー番号のエラー・メッセージを CRT に表示し実行中止。
- 3 ユーザ独自のエラー番号を定義する場合、システムで使用されていない番号を使う。その場合、ON ERROR GOTO 文があると、指定行へ制御が移る。
- 4 ON ERROR GOTO 文のデバッグなどに使うと便利である。使用者が独自のエラー・メッセージを作るのにも使う。

プログラム例: エラー番号 36 のエラーを模擬的に発生させる。

```

100 ON ERROR GOTO 200
110 A=13:A1=23
120 ERROR A+A1
130 PRINT "END"
140 END
200 PRINT "ERROR NUMBER= ";ERR
210 PRINT "ERROR LINE  = ";ERL
220 RESUME NEXT

```

実行例

```

ERROR NUMBER= 36
ERROR LINE  = 120
END

```

参照: ON ERROR GOTO, ERL, ERR

APL —

TRS_I 算術式の値は整数で 1~255 の範囲。
・CRT にエラー・コードを表示し、実行中止する。

CPM 本文と同じ。

M223 —

MZK —

CBM —

PC8 本文参照。

TRS_{II} TRS_I と同じ。

PC3 —

IF8₂ ERROR 3

- ・数値は 1~255 の範囲。エラー番号としてシステムで使用されている以外の数値の場合、"Unprintable error" を出力する。
- ・ERROR 文を実行すると ERL 関数の値は ERROR 文のある行番号となる。

レベル₃ 数値式の値は 1~255 の範囲の整数。

- C180 数値式の値は整数で 1~255 の範囲。
・主プログラム中の ERROR 文を実行すると数値式の値を CRT に表示して実行中止。
・副プログラム中の ERROR 文を実行すると CALL 文にエラーが発生したものとし、CALL の次へもどり、エラー処理部へ移るか、そうでなければ実行中止か、次の文から続行する。
・ユーザによる、エラー番号を定義

したいときは、CAUSE A1+3 を用いる。

M243 —

MZB —

BUB 算術式の値の範囲は 1~255。

FM8 BUB と同じ。

PSP —

PC88 本文と同じ。

N52 IF8₂ と同じ。

PC6 —

MLT 本文と同じ。

HC IF8₂ と同じ。FP11 TRS_I と同じ。

SMC 整数表記は 1~255 の範囲。
・ERR (N) でも使える。

MZ35 —

X1Hu IF8₂ と同じ。ただし、ERROR 文を実行すると ERR 変数にエラー・コード、ERL 変数にその行番号が入る。
・引数は整数に四捨五入される。

MB16 本文と同じ。

IBM55 本文と同じ。

PC100 本文と同じ。

MSX BUB と同じ。

入出力エラー分岐指定 TRAP ON

トラップ・オン

形式：TRAP ON

↑
必要

機能：入出力エラーが発生したとき、エラー処理部へ分岐するか否かを指定する。

特徴：1 TRAP ON で分岐指定する。
2 TRAP OFF で分岐指定を解除する。
3 エラー処理部の範囲は EXCEPTION と EXCEPTION END で指定する。

プログラム例：I/O エラーが発生した場合、そのファイル番号と、エラー・コード（FILE関数の値）を出力し、EOFの場合、ファイルを CLOSE して終わる。その他のエラーの場合、そのエラー番号を出力する。行番号140でのエラーは、B(1)が0の場合なのでB(1)を1にしてその行を再実行する。

```

100 DIM B(1)
110 TRAP ON
120 OPEN #5:"BFILE",INPUT
130 INPUT #5:B(1)
140 PRINT 1/B(1)
150 GOTO 130
160 CLOSE #5
170 EXCEPTION SECTION
180 IF EXNO=1 THEN
190   IF FILE(EXFNO)=7 THEN      実行例：(その1)
200     RESUME 160              I/O エラーの場合。
210   END IF
220   PRINT "I/O ERR"
230   PRINT "FILE NO =" ; EXFNO  I/O ERR
240   PRINT "ERR CODE =" ; FILE(EXFNO) FILE NO =-1
250   RESUME 350                 ERR CODE= 3
260 ELSE
270   IF EXLINE=140 THEN
280     B(1)=1
290     RETRY                    実行例：(その2)
300   END IF                    I/O エラー以外の
310   PRINT "ERR =" ; EXNO      エラーの場合。
320   RESUME 350               ERR = 52
330 END IF
340 EXCEPTION END
350 END

```

参照：EXCEPTION SECTION, EXCEPTION END, ON ERROR GOTO

APL ・ ONERR GOTO 70 で
TRAP ON と同様のことができる。TRS_I ・ ON ERROR GOTO 70
で、TRAP ON, ON ERROR
GOTO 0 で、TRAP OFF と同様の
ことができる。CPM ・ TRS_I と同じ。

M223 ・ APL と同じ。

MZK ・ APL と同じ。

CBM —

PC8 ・ TRS_I と同じ。TRS_{II} ・ TRS_I と同じ。PC3 ・ ON ERROR GOTO 70
または、ON ERROR GOSUB
70 で TRAP ON, OFF ERROR
で TRAP OFF と同様のことがで
きる。IF8₂ ・ TRS_I と同じ。レベル₃ ・ TRS_I と同じ。

C180 ・ 本文参照。

M243 ・ APL と同じ。

MZB ・ APL と同じ。

BUB ・ TRS_I と同じ。FM8 ・ TRS_I と同じ。

PSP —

PC88 ・ TRS_I と同じ。N52 ・ TRS_I と同じ。

PC6 —

MLT ・ TRS_I と同じ。HC ・ TRS_I と同じ。

FP11 —

SMC ・ ERROR ON で、TRAP
ON, ERROR OFF で、TRAP
OFF と同様なことができる。

MZ35 ・ PC3 と同じ。

X1Hu ・ TRS_I と同じ。MB16 ・ TRS_I と同じ。

IBM55 —

PC100 —

MSX ・ TRS_I と同じ。

エラー処理後の再開 RESUME

リジューム

形式: RESUME 70

省略可

機能: エラー処理の終了後、プログラムの実行を再開する。

- 特徴: 1 エラー処理の実行後、プログラムの実行を指定した行番号の行から実行する。
- 2 実行を再開する場所によって以下の形式を選ぶ。
- (a) RESUME または RESUME 0
エラーの発生した文から、プログラムの実行を再開する。
- (b) RESUME NEXT
エラーの発生した文の次の文から、プログラムの実行を再開する。
- 3 RESUME 文は、エラー処理プログラムの終わりを示す。
- 4 エラー処理後、プログラムの実行の再開に使われる。

プログラム例: DATA 文のデータを複数回 READ 文で読み込む。

```

10 ON ERROR GOTO 100
20 DIM B(10)
30 FOR I=0 TO 10
40 READ B(I)
50 NEXT I
60 FOR I=0 TO 10:PRINT B(I);:NEXT I
70 DATA 0,1,2,3,4,5,6,7
80 END
100 IF ERR = 4 THEN RESTORE
110 RESUME 40

```

実行例

0 1 2 3 4 5 6 7 0 1 2

参照: ERR, ERL, ON ERROR GOTO

APL ・本文と同じ。ただし、行番号の指定は行えず、エラーの発生した文の最初から実行を再開する。

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。ただし、行番号を 0 または省略したときは、ERR=0 とし、エラーの発生した次の文から実行を再開する。

MZK ・本文と同じ。ただし、行番号を 0 としたときは、プログラムの先頭から実行を再開する。

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・エラー処理の終了後は、RETURN または GOTO 文を用いて実行を再開する。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。ただし、行番号の指定を 0 にはできない。C180 ・M223 と同じ。ただし、エラー処理部外でも引用できる。
・エラーが発生した行へもどる
RETRY もある。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・レベル₃ と同じ。

PSP —

PC88 ・行番号の代わりにラベル名も使える。

N52 ・2 の (a) は RESUME のみ有効。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・レベル₃ と同じ。

SMC ・PC88 と同じ。ただし、RESUME0 は、文番号 0 の行から再開する。

MZ35 ・ON で始まるステートメントでジャンプしたサブルーチンを終わらせ、処理の流れをもどす。

X1Hu ・SMC と同じ。また、ラベル名も使える。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・PC88 と同じ。

MSX ・本文と同じ。

エラー発生行 ERL

エラー・ライン

形式: ERL

機能: エラーが発生したとき、その行番号が設定される。

- 特徴:
- 1 IF~THEN 文で通常用いられる。
 - 2 この変数に代入はできない(予約語)。
 - 3 エラーの原因となった文が直接文であるかを調べるには、IF ERL=65535 THEN...を用いる。
 - 4 ERLの値は、ERRの値が0以外のとき意味をもつ。
 - 5 直接実行文以外の場合は、ERLはリナンバの対象となるので、条件式での行番号は右辺にしなければならない。
 - 6 エラー処理において、エラーの原因を調べるのに便利である。

プログラム例: 平方根と対数を求める。

```

10 ON ERROR GOTO 100
20 INPUT A
30 PRINT "SQR( ";A;" )=";SQR(A),
40 PRINT "log( ";A;" )=";LOG(A)
50 GOTO 20
100 IF ERL=30 THEN 130
110 IF ERL=40 THEN PRINT " ERROR"
120 RESUME 20
130 PRINT SQR(-A);"i",:RESUME 40

```

実行例

```

? 5
SQR( 5 )= 2.23607      log( 5 )= 1.60944
? -5
SQR(-5 )= 2.23607 i    log(-5 )= ERROR
?
Break in 20

```

参照: ERR, ON ERROR GOTO, RESUME, ERROR, CAUSE

APL —

TRS_I ・本文と同じ。ただし、エラーのない状態では、ERLの値は0。

CPM ・本文と同じ。

M223 ERRL
 ・機能は本文と同じ。ただし、ON ERROR GOTO文が、エラー発生前に実行されてないときは、ERRLには文番号は設定されない。
 ・ERRLの値は1~32255の範囲。

MZK ・本文と同じ。

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 —

IF₈₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 EXLINE
 ・機能は本文と同じ。ただし、エラー処理部の外でも引用できる。
 ・初期状態とエラー処理部からもどるときEXLINEの値は-1。

M243 ・M223と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・本文と同じ。

MZ35 ・本文と同じ。なお、特徴2で代入することができる。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

エラー処理 ERR

エラー・ナンバ

形式: ERR

機能: エラーが発生したとき, そのエラー番号が設定される。

- 特徴:
- 1 IF~THEN 文で通常用いられる。
 - 2 この変数に代入はできない (予約変数)。
 - 3 エラーのない状態では, ERR の値は 0。
 - 4 エラー処理において, エラーの原因を調べるのに便利である。

プログラム例: 逆数を求める (0で除算を行うとエラー, エラー番号 11 が設定される)。

```

10 ON ERROR GOTO 100
20 INPUT A
30 PRINT "1/" ; A ; " = " ; 1/A
40 GOTO 20
100 IF ERR=11 THEN PRINT "ERROR"
110 RESUME 20

```

実行例

```

? 5
1/ 5 = .2
? 0
1/ 0 = ERROR
? -5
1/-5 = -.2
?
Break in 20

```

参照: ERL, ON ERROR GOTO, RESUME, PEEK, ERROR, CAUSE

APL ・エラー番号は 222 番地に格納されるので, PEEK (222) によりエラー番号が得られる。

TRS₁ ・本文と同じ。ただし、
((ERRの値)/2)+1 がエラー番号を示す。

CPM ・本文と同じ。
・ST も参照。

M232 ・本文と同じ。

MZK ERN
・機能は本文と同じ。

CBM DS
・DOS のエラー番号を示す。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 EXNO
・機能は本文と同じ。ただし、エラーのない状態では EXNO の値は -1。

M243 ・本文と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・本文と同じ。

MZ35 ・MZK と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

副プログラムへの分岐 GOSUB

ゴ-サブ

形式：GOSUB 70

文番号

機能：サブルーチンへの分岐。

- 特徴：
- 1 行番号は、サブルーチンの最初の行番号を指定する。
 - 2 サブルーチンの中で、別のサブルーチンを呼ぶことができる(ネスティング)。
 - 3 GOSUB のうしろは、文番号に限る(算術式は使えない)。
 - 4 プログラム中で同じ作業を幾度か行う場合に、その作業を一つにまとめて、サブルーチンとすると、便利である。

プログラム例：サブルーチン (60~70) を実行する。

```

10 I=3
20 GOSUB 60
30 LET I=I*2
40 LPRINT "I=", I
50 END
60 LPRINT "SUB 60 I=", I
70 RETURN

```

実行例

```

SUB 60  I=      3
I=      6

```

参照：RETURN, GOTO, ON GOSUB

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ 本文と同じ。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 GOSUB "LABEL"
GOSUB (A1+2)
・ 行番号のほかにラベル名と数式を使用できる。数式の値は小数点以下切り捨てられる。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 GOSUB 70
または、GO SUB 70

M243 ・ 本文と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 GOSUB 70
GOSUB *LABEL
・ 行番号のほかにラベル名も使える。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 GOSUB 70
GOSUB PROG1
・ 行番号のほかに、プログラム・エリア番号を指定できる。

SMC ・ PC88 と同じ。

MZ35 ・ PC3 と同じ。

X1Hu ・ 行番号の代わりにラベル名も使える。
・ GO SUB でもよい。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ PC88 と同じ。

MSX ・ 本文と同じ。

複数サブルーチンへの分岐 ON GOSUB

オン・ゴー・サブ

形式：ON A1+3 GOSUB 70, 170, 270

算術式

省略、繰り返し可

機能：式の値を計算し、その結果によって与えられた幾つかの行番号を先頭とするサブルーチンにプログラムの制御を移す。

特徴：

- 1 式の値の小数部分は切り捨てられる。
- 2 式の値が負の場合はエラーとなる。
- 3 式の値が0または行番号の個数より大きい場合は、プログラムの制御は次の行へ移り、エラーは起こらない。
- 4 プログラム制御の流れを幾つかに分岐させ、再び一つにまとめる場合に便利である。

プログラム例：Iの値を1として、式I+2の値をみて、サブルーチン(80~90)へ分岐する。

```
10 I=1
20 ON I+2 GOSUB 40,60,80
30 END
40 LPRINT "SUB-1"
50 RETURN
60 LPRINT "SUB-2"
70 RETURN
80 LPRINT "SUB-3"
90 RETURN
```

実行例

SUB-3

参照：ON GOTO, RETURN

APL ・本文と同じ。

TRS_I ・本文と同じ。

CPM ・本文と同じ。ただし、式の値の小数部分は四捨五入される。式の値が0または行番号の個数より大きい場合でもエラーが起こり、メッセージが表示される。

M223 ・本文と同じ。ただし、式の値が負の場合でもエラーにならず、次の行にプログラムの制御が移る。

MZK ・M223と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。さらに、行番号の代わりにラベル名が使用できる。

IF8₂ ・CPMと同じ。

レベル₃ ・式の値の小数部分は四捨五入される。

C180 ON A1+3 GOSUB 70, 170 ELSE 実行文
・小数部分は四捨五入。式の値は0以下、または行番号の個数より大きい場合、ELSEの次の実行文を実行する。ELSEがないときは、エラー表示の“E108”がある。

M243 ・M223と同じ。

MZB ・M223と同じ。

BUB ・CPMと同じ。

FM8 ・M223と同じ。

PSP ・M223と同じ。

PC88 ・PC3と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・特徴1で、小数部分は四捨五入される。また、特徴2では、負の場合と256以上の場合にエラーとなる。

HC ・算術式の値は0~255の範囲でなくてはならない。

FP11 ・M223と同じ。また、行番号のほかに、プログラム・エリア番号を指定できる。

SMC ・PC3と同じ。ただし、式の値が256以上の場合もエラーとなる。

MZ35 ・PC3と同じ。なお、特徴2はM223と同じ。

X1Hu ・レベル₃と同じ。なお、行番号の代わりにラベル名も使える。

MB16 ・レベル₃と同じ。

IBM55 ・MLTと同じ。

PC100 ・小数部分は四捨五入される。
・ラベル名も使える。

MSX ・本文と同じ。

副プログラムからのもどり RETURN

リターン

形式：RETURN

機能：プログラムの制御を、サブルーチンからメイン・プログラムにもどす。

特徴：1 一つのサブルーチンに二つ以上の RETURN 文があってもよい。

プログラム例：I×Iの(*)からなる正方形を書かせるサブルーチンをI=2, 4について呼ぶ。

```

10 I=2
20 GOSUB 60
30 I=4
40 GOSUB 60
50 END
60 REM ---SUBROUTINE---          実 行 例
70 FOR J=1 TO I
80 LPRINT STRING$(I, "*")
90 NEXT J
100 LPRINT
110 RETURN

```

APL ・一つのサブルーチンに一つしか使用できない。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。

IF8₂ ・BUBと同じ。

レベル₃ ・本文と同じ。

C180 ・本文と同じ。

M243 ・本文と同じ。

MZB ・本文と同じ。

BUB RETURN 70
・行番号(省略可)がある場合、その行へもどる。

FM8 ・BUBと同じ。

PSP ・本文と同じ。

PC88 ・BUBと同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・BUBと同じ。

HC ・本文と同じ。

FP11 ・BUBと同じ。また、行番号のほかに、プログラム・エリア番号を指定できる。その場合、指定プログラム・エリアの先頭へもどる。

SMC ・本文と同じ。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。さらに、行番号またはラベル名があるとその行へもどる。

MB16 ・BUBと同じ。

IBM55 ・本文と同じ。

PC100 ・BUBと同じ。

MSX ・本文と同じ。

参照：GOSUB, SUBEXIT

上位ルーチンへのもどり POP

形式：POP

機能：ルーチンAがサブルーチンBを呼び、Bの中でまたサブルーチンCを呼んでいるとき、サブルーチンCから処理を直接ルーチンAへもどす。

特徴：1 RETURN文の前にPOP文をおくことによって、中間のサブルーチンを飛びこしてその上位ルーチンへ処理をもどせる。
2 サブルーチンの中で、エラーなどの例外的なことが発生したとき、直接メイン・ルーチンへもどりたときなど便利である。

プログラム例：割り算を実行するサブルーチンで、分母が0であったら、上位のルーチンへもどる。

```

10 INPUT "*", / OR S(STOP) ?"; C#
20 IF C#="S" THEN END
30 IF C#="/" THEN GO SUB "/":ELSE GO SUB "*"
40 GO TO 10
50 "*" / : INPUT "DIVISION, INPUT A,B"; A,B
60 PRINT A;" / "; B;
70 GO SUB "KEISAN"
80 PRINT "KEKKA="; D
90 RETURN
100*"KEISAN": IF B<>0 THEN GO TO "C"
110 PRINT "DIVIZOR ZERO!": POP : RETURN
120*"C": D=A/B: RETURN
130*"*": RETURN

```

実行例

```

10 / 20      KEKKA= 0.5
10 / 0       DIVIZOR ZERO!

```

参照：RETURN

APL ・ 本文と同じ。

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 ・ 本文参照。

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ・ PC3と同じ。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

副プログラムの呼び出し CALL

コール

形式: CALL ASUB (A, #9)

↑
必要

*省略・繰り返し可

省略可

機能: 指定した副プログラムを呼び出す。

- 特徴:
- 1 副プログラム名は、8文字以内の英数カナ文字列。
 - 2 実引数¹⁾は変数、算術式、配列引数 (副プログラムに配列名を渡すときに使う)、ファイル番号とする。
 - 3 実引数は8個以内である。
 - 4 システム2進数値変数 (CRT%) は、実引数とはできない。
 - 5 引数をもたない関数を、単独で実引数として使用する場合、カッコで囲まなければならない。
 - 6 PLOAD文などで格納した、主プログラム内にはない副プログラムを呼び出すのに CALL 文は必要である。
 - 7 副プログラム名は、Zで始められない。
 - 8 実引数の値を変えない場合は、カッコで囲まなければならない。
 - 9 C180では独立した副プログラムを作成できるので便利である。そのような副プログラムを呼び出すのに使う。

用語: ¹⁾実引数 [C180] 主プログラムが副プログラムを呼んだとき、主プログラム側からみて渡す引数または受けとる引数。

プログラム例: N×N個の(＃)からなる正方形を書かせる副プログラムを、入力したNについて実行する。

主プログラム

```
10 PLOAD "SQR.IB/0"
20 INPUT N
30 PRINT "N=";N
40 CALL SQR(N)
50 END
```

実行例

副プログラム

```
10 SUB SQR(P1)
20 IF P1<=0 THEN SUBEXIT
30 FOR I=1 TO P1
40 PRINT REP$( "#",P1)
50 NEXT I
60 SUBEXIT
70 SUBEND
```

```
? 5
N= 5
#####
#####
#####
#####
#####
```

参照: SUB, SUBEXIT, SUBEND, PLOAD

APL —

TRS₁ —

CPM —

M223 —

MZK ・該当なし。ただし、SWAP文によってディスクット・ファイルのプログラムを呼び出し、実行した後SWAP文の次へもどることができる。

CBM —

PC8 —

TRS₁₁ —

PC3 ・本文と同じ。ただし、実引数としてファイル番号は使えない。

IF8₂ —レベル₃ —

C180 ・本文参照。

M243 —

MZB ・MZKと同じ。

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ・本文と同じ。なお、特徴3で8個以上使え、特徴7でZも使える。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX ・該当なし。ただし、ROMカートリッジ内の拡張ステートメントを、本文の形式により実行させることができる。

副プログラムの先頭 SUB

形式: SUB ASUB (A, B ())

↑
必要*省略・繰
り返し可

省略可

機能: 副プログラムの先頭を示す。

- 特徴: 1 仮引数¹⁾には変数、配列引数、ファイル番号とする。ただし、ファイル番号は、整数の数値定数に限る。引数は7個まで。
 2 配列引数は、呼び出し元の配列を示す。したがって、副プログラム中で宣言してはならない。
 3 仮引数で指定していない変数やファイル番号は、副プログラムのみで有効となる。
 4 システム2進数値変数は、仮引数に使えない。ただし、主・副プログラムに関係なく共通である。
 5 プログラムの先頭にこれをつけることにより、主プログラムと独立に副プログラムを作ることができる。
 6 実引数と仮引数の個数、型、次元は、一致しなければならない。
 7 配列引数のときは、上記のB()のように()をつける。2次元のときはB(,)。

用語: ¹⁾仮引数 [C180] 副プログラムが主プログラムに呼ばれた時、副プログラム側からみて受けとるまたは渡す引数。

プログラム例: N×N個の(＃)からなる正方形を書かせる副プログラムを、入力したNについて実行する。

主プログラム

```
10 PLOAD "SQR.IB/0"      実行例
20 INPUT N                ? 9
30 PRINT "N=";N           N= 9
40 CALL SQR(N)            #####
50 END                    #####
```

副プログラム

```
10 SUB SQR(P1)            #####
20 IF P1<=0 THEN SUBEXIT #####
30 FOR I=1 TO P1          #####
40 PRINT REP$( "#",P1)    #####
50 NEXT I                 #####
60 SUBEXIT                #####
70 SUBEND
```

参照: CALL, SUBEXIT, SUBEND

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 ・本文と同じ。ただし、ファイル番号は使えない。また配列引数はB(*)。

IF8₂ —レベル₃ —

C180 ・本文参照。

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ・PC3と同じ。なお、特徴1で7個以上使え、引数B(*)とすると3次元以上も可能。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

副プログラムの終了 SUBEND

サブ・エンド

形式：SUBEND

機能：副プログラムの終わりを示す。

特徴：1 呼び出し元へもどる働きももっている。

プログラム例：N×N個の（#）からなる正方形を書かせる，副プログラムを，入力したNについて実行する。

主プログラム

```
10 PLOAD "SQR.IB/0"
20 INPUT N
30 PRINT "N=";N
40 CALL SQR(N)
50 END
```

副プログラム

```
10 SUB SQR(P1)
20 IF P1<=0 THEN SUBEXIT
30 FOR I=1 TO P1
40 PRINT REP$( "#",P1)
50 NEXT I
60 SUBEXIT
70 SUBEND
```

実行例

```
? 4
N= 4
####
####
####
####
```

参照：CALL, SUB, SUBEXIT

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 ・本文と同じ。

IF8₂ —レベル₃ —

C180 ・本文参照。

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FPI1 —

SMC —

MZ35 ・PC3と同じ。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

副プログラムの出口 SUBEXIT

サブ・イグジット

形式：SUBEXIT

機能：呼び出した CALL 文の次の行へもどる。

特徴： 1 副プログラムの中にだけ書くことができる。
 2 SUBEND と共に使用する必要がある。

プログラム例：N×N個の(+)からなる正方形を書かせる副プログラムを、入力したNについて実行する。

主プログラム

```
10 PLOAD "SQR.IB/0"
20 INPUT N
30 PRINT "N=";N
40 CALL SQR(N)
50 END
```

副プログラム

```
10 SUB SQR(P1)
20 IF P1<=0 THEN SUBEXIT
30 FOR I=1 TO P1
40 PRINT REP$( "+", P1)
50 NEXT I
60 SUBEXIT
70 SUBEND
```

実行例

```
? 5
N= 5
+++++
+++++
+++++
+++++
+++++
```

参照：CALL, SUB, SUBEND

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 SUBEND
 ・機能は本文と同じ。

IF8₂ —レベル₃ —

C180 ・本文参照。

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ・PC3と同じ。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

機械語ルーチンの呼び出し USR

ユーザ

形式: USR M (A)

省略可

機能: 引数 A をもって、ユーザの機械語ルーチンを呼び出す。

特徴: 1 引数 M は、DEFUSR 文で定義した番号に対応し、0 から 9 までの整数である。

2 M を省略すると、USR 0 となる。

プログラム例: E000 番地の機械ルーチン (引数個の * を表示する) を呼び出し実行する。

```

10 DEF USR2=&HE000
20 FOR I=&HE000 TO &HE00C
30 READ A$
40 POKE I,VAL("&h"+A$)
50 NEXT I
60 A=USR2(5)
70 PRINT
80 B=USR2(10)
90 END
100 DATA eb,1a,47,3e,2a,cd,57,02,05,c2,05,e0,c9

```

実行例

```

*****
*****

```

APL USR (A)

・機能は同じだが、特徴 1, 2 の機能はない。

TRS_i ・本文と同じ。ただし、
-32768 ≤ A ≤ 32768。

CPM ・本文と同じ。

M223 CALL #n, A1, B...

- ・0 ≤ n ≤ 31, 引数最大 8 個で単精度の算術式。
- ・機械語ルーチンは BASIC とリンクして新しい BASIC を作成してからでないと使えない。

MZK USR (AD)

・AD は 10 進番地または &16 進番地で指定番地である。

CBM ・APL と同じ。

PC8 ・本文参照。

TRS_{II} ・TRS_i と同じ。

PC3 USR AD, A (A は省略可)

- ・引数 A で、メモリ番地 AD (10 進または &16 進) から始まる機械語ルーチンを呼び出す。ただし、関数ではない。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 —

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・該当なし。ただし、EXEC 参照。

PC88 ・本文と同じ。

N52 —

PC6 ・FAEB 番地と FAEC 番地に、機械語プログラムの先頭アドレスの下位 8 ビット、上位 8 ビットを POKE 文で書き込むことで、ユーザの機械語プログラムの開始番地を PC6 に知らせる。

・BASIC プログラムから機械語プログラムにデータを受け渡すには、機械語プログラムで 741 番地をコールする。USR (X)

MLT ・機械語ルーチンを BASIC プログラム領域と異なるセグメントにおく場合、USR 文の前に必ず DEF SEG 文の実行が必要。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ANN M (A)

- ・M は DEF ANN 文で定義した番号に対応し、0 から 9 までの整数。
- ・機能は本文と同じ。

MZ35 ・PC3 と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。また、CALL 文を用いることもできる。

IBM55 ・MB16 と同じ。

PC100 ・MB16 と同じ。

MSX ・本文と同じ。

参照: EXEC, DEFUSR

指定メモリからの直接読み出し PEEK

形式: PEEK (M)

アドレス

機能: 指定されたメモリの内容 (0 から 255 までの値) を読む。

特徴: 1 アドレスは, 0 から 65535 までの範囲。
2 PEEK は, POKE の逆の働きをする関数である。用語: ¹メモリ・バンク [IF8₂] メモリ中に直接書き込むことのできる場所のことで、VRAM も含めたものをいう。メイン・メモリに一つと VRAM の三つから構成されている。

プログラム例: &H8000 番地から &H8010 番地にある, A から Q までのアスキー・コードを直接読み出す。

```

10 FOR I=&H8000 TO &H8010
20 POKE I,&H41+(I-&H8000)
30 NEXT I
40 FOR J=&H8000 TO &H8010
50 LPRINT CHR$(PEEK(J));
60 NEXT J
70 END

```

実行例

ABCDEFGHIJKLMNO P Q

参照: POKE

APL ・本文と同じ。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。ただし、アドレスの指定は数式でもよい。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS₁₁ —

PC3 ・M223 と同じ。

IF8₂ PEEK (M, 2)

- ・機能は本文と同じ。
- ・メモリ・バンク¹指定 (2 番目の引数) は, 1 から 7 までの範囲。
- ・メモリ・バンク指定がないとき, メイン・バンクが対象となり, メモリ・バンク指定があるとき, アドレスは 32768 から 49151 までの範囲。

レベル₃ ・本文と同じ。

C180 —

M243 ・M223 と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 —

PC6 ・本文と同じ。

MLT ・M の値は, DEF SEG 文で与えられた現在のセグメントからのオフセット値を示す。

HC ・本文と同じ。

FP11 ・M の範囲は, -32769 < M < 65536。

SMC ・M223 と同じ。ただし, -32769 < M < 65536。
VINP (M)
・V RAM から直接読み出しをする。

MZ35 ・M223 と同じ。ただし, アドレスは 38400 から 49151 の範囲。

X1Hu ・本文と同じ。
PEEK@ (&H2000)
・VRAM の内容の読み出しもできる (引数は &H2000 ~ &HFFFF)。

MB16 ・MLT と同じ。

IBM55 ・MLT と同じ。

PC100 ・MLT と同じ。

MSX ・本文と同じ。ただし, -32769 < M < 65536。
VPEEK (N)
・VRAM から直接読み出しをする。
VDP (N)
・VDP (ビデオ・ディスプレイ・プロセッサ) レジスタ内容の読み出し, 書き込みを行う。
BASE (N)
・VDP テーブルのベース・アドレスの書き込み, 読み出しを行う。

指定メモリへの直接書き込み POKE

ポーク

形式: POKE M, N

└─┘
└─┘
 アドレス データ

機能: 指定されたメモリに、データを書き込む。

- 特徴:
- 1 アドレスは 0 から 65535 までの範囲。
 - 2 データは、0 から 255 までの範囲。
 - 3 POKE の逆の働きをする関数に、PEEK がある。
 - 4 POKE と PEEK は、効率的なデータの格納や、アセンブル言語のサブルーチンのロードや、アセンブル言語のサブルーチンとの引数や結果の受け渡しなどに用いると便利である。
 - 5 書き込めないアドレスもあるので注意。

プログラム例: &H8000 番地から、&H8010 番地に A から Q までのアスキー・コードを直接書き込む。

```

10 FOR I=&H8000 TO &H8010
20 POKE I,&H41+(I-&H8000)
30 NEXT I
40 FOR J=&H8000 TO &H8010
50 LPRINT CHR$(PEEK(J));
60 NEXT J
70 END

```

実行例

ABCDEFGHIJKLMNO P Q

参照: PEEK

APL ・本文と同じ。

TRS_I ・本文と同じ。

CPM ・本文と同じ。ただし、メモリは、0 から 65536 までの範囲。

M223 ・本文と同じ。ただし、アドレス、データの指定は数式でも書ける。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} —

PC3 POKE M, N1, N2
 ・機能は本文と同じ。
 ・アドレスは、19712 から 53247 までの範囲。
 ・データを N1, N2, … と繰り返したとき、アドレスは M, M+1, M+2, … となる。

IF8₂ POKE M, N, 3
 ・機能は本文と同じ。
 ・メモリ・バンク指定 (3 番目の引数) は、1 から 7 までの範囲。
 ・メモリ・バンク指定がないとき、メイン・バンクが対象となり、メモリ・バンク指定があるとき、アドレスは 32768 から 49151 の範囲。

レベル₃ ・本文と同じ。

C180 —

M243 ・M223 と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。ただし、PO M, N と省略可。

PC88 ・本文と同じ。

N52 —

PC6 ・本文と同じ。

MLT ・M の値は、DEF SEG 文で与えられたセグメント番地からのオフセット値を示す。

HC ・本文と同じ。

FP11 ・M の範囲は、-32769 < M < 65536。

SMC ・M223 と同じ。ただし、-32769 < M < 65536。
 VOUT M, N
 ・V RAM にデータを出力する。

MZ35 ・メモリ・アドレスは 38400 から 49151 までの範囲。

X1H_u POKE M, N1, N2, …
 ・機能は本文と同じ。
 ・データを N1, N2, … と繰り返したとき、アドレスは M, M+1, M+2, … となる。
 POKE@ M, N1, N2 …
 ・VRAM への書き込み (アドレスは &H2000 ~ &HFFFF)。

MB16 ・MLT と同じ。

IBM55 ・MLT と同じ。

PC100 ・MLT と同じ。

MSX ・本文と同じ。ただし、-32769 < M < 65536。

指定メモリへの直接書き込み POKE

<p>VPOKE (N)</p> <ul style="list-style-type: none"> ・ VRAM に直接データを書き込む. <p>VDP (N)</p> <ul style="list-style-type: none"> ・ VDP レジスタの読み出し, 書き込みを行う. <p>BASE (N)</p> <ul style="list-style-type: none"> ・ VDP テーブルのベース・アドレスの読み出し, 書き込みを行う. 			
--	--	--	--

未使用バイト数の表示 FRE

フリー

形式：FRE (A)

機能：メモリの未使用バイト数を与える関数。

- 特徴：1 引数はグミー。
 2 引数が数値の場合は、BASIC の使っていないメモリのバイト数を与える。
 3 引数は文字も可能で、その場合は、文字領域の未使用バイト数を与える。
 4 未使用メモリの大きさを見ることにより、使用したメモリの大きさがわかるので、大きなメモリの必要なプログラムを入力するときなど便利である。

プログラム例：BASIC の使っていないメモリの大きさと、文字領域の未使用バイト数を印字させる。

```
10 LPRINT FRE(5)
20 LPRINT FRE("F")
30 END
```

実行例

```
20394
300
```

参照：

APL ・本文と同じ。ただし、特徴3の機能はない。

TRS_I ・APLと同じ。

CPM ・本文と同じ。

M223 ・コマンドで SIZE が同じ役目をする。引数はない。

MZK ・M223 と同じ。

CBM ・APLと同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。PC3 STATUS (2)
 ・ユーザ・エリアの未使用バイト数を表示。IF8₂ ・APLと同じ。ただし、A が "" のときは未使用メモリの大きさを与える。レベル₃ ・本文と同じ。

C180 —

M243 ・M223 と同じ。

MZB ・M223 と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・APLと同じ。

PC88 ・本文と同じ。ただし、引数Aが0ならば、未使用変数領域、つまり、単純変数、配列およびストリングの領域として使用可能な領域の残りのバイト数を表示し、

1 ならば、未使用テキスト領域、つまり、プログラム・テキストを入れるための領域の残りバイト数を表示、2 ならば、未使用変数領域と未使用テキスト領域の和のバイト数を表示する。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC STAT ALL
 ・RAM ファイルの大きさと MEMSET のアドレス、未使用テキスト領域の大きさを表示。
 ・ALL の代わりにプログラム・エリア番号 1 ～ 5 を入れると、各エリアに格納されているプログラムの名前と大きさを表示する。
 ・これに関連して、RAM/ROM パックの未使用領域の大きさを示す、PACF 関数がある。

FP11 ・本文と同じ。

SMC ・A が任意のデータ、または、省略された時、文字列関数の古いデータ部分も使用領域として未使用バイト数を返し、A がナルストリングの時は、文字列関数の古いデータを消去して未使用バイト数を返す。

MZ35 ・PC3 と同じ。

X1Hu ・APL と同じ。なお、SIZE でも同じ役目をする。

MB16 ・A が文字の場合、未使用メモリ領域のバイト数を返すと同時に有効なデータを集約し、一度使用された現在有効でない領域を使えるようにする。

変数の番地表示 VARPTR

バリアブル・ポインタ

形式: VARPTR (A)

機能: 変数が入っている主記憶領域上の番地を得る。

- 特徴:
- 1 変数はあらかじめ値が代入されていなければエラーが発生する。
 - 2 変数としては整数、単精度、倍精度、文字列各変数と配列、ファイル番号が使用できる。
 - 3 関数値は32767から-32768までの範囲。
 - 4 変数にファイル番号を使用すると、値はそのファイルに割り付けられている入出力バッファの開始番地となる。
 - 5 アセンブリ言語のサブルーチンに移る場合、変数または配列の番地を調べるときに使うと便利である。

用語: "セグメント" メモリのひとつのかたまりをいう。

プログラム例: 変数 A, B\$ の格納番地を調べる。

```

10 A=22.5
20 B$="ABC"
30 PRINT "ADDRESS OF 'A' = ";VARPTR(A)
40 PRINT "ADDRESS OF 'B$' = ";VARPTR(B$)
50 END

```

実行例

```

ADDRESS OF 'A' = -32636
ADDRESS OF 'B$' = -32629

```

参照: OPEN

APL —

TRS_i ・本文と同じ。ただし、変数に配列とファイル番号は使用できない。

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・TRS_i と同じ。

PC3 —

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・該当なし。ただし、VARPTR% (A) で同様のことが可。関数値は2進数値で、ファイル番号を変数として使用できない。

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。ただし、変数にファイル番号は使用できない。

PSP ・該当なし。ただし、ADR (A) で同様のことが可。

PC88 ・本文と同じ。

N52 —

PC6 —

MLT ・VARPTR 関数を与える番地は、0～65535の整数値で、DEF SEG 文で与えられた、現在のセグメント¹⁾番地からのオフセット値で示す。
・ファイル番号を指定するときは、VARPTR (#5) で表示。この場合、そのファイルに対して設けられている、ファイル制御情報領域の先頭番地を表示する。

HC ・本文と同じ。

FP11 ・特徴1はない。また、変数として倍々精度も使用できる。
・これに関連して、使用変数のリストを表示する VARLIST 命令がある。

SMC ・本文と同じ。

MZ35 —

X1Hu ・本文と同じ。ほかに、STRPTR 関数がある。

MB16 ・本文と同じ。

IBM55 ・MLT と同じ。その他にアドレスの文字型式を返す VARPTR \$ 関数がある。

PC100 ・アドレスをアスキー型式で与えるものとして VARPTR \$ がある。

MSX ・本文と同じ。

機械語プログラムの実行 EXEC

エグゼキュート

形式: EXEC &H7000

省略可

機能: 機械語プログラムを実行する。

- 特徴: 1 指定した開始番地から、機械語プログラムを実行する。
 2 開始番地が省略されたときは、直前に実行された LOADM コマンドの入口番地、または直前の EXEC コマンドの開始番地から実行する。

プログラム例: プリントアスキー・コード 40₁₆~5F₁₆ の文字を印
 字する。

```
10 CLEAR 100, &H6FFF
20 FOR I=&H8000 TO &H8025
30 READ A#
40 POKE I, VAL("&H"+A#)
50 NEXT I
60 EXEC &H8000
70 DATA 86, 20, B7, 80, 30, 86, 3F, B7
80 DATA 80, 32, CE, FF, C2, 7C, 80, 32
90 DATA B6, 80, 32, BD, ED, 71, 7A, B0
100 DATA 30, 27, 02, 20, ED, 86, 0A, BD
110 DATA ED, 71, 39, 00, 00, 00
120 END
```

実行例

```
0ABCDEF GHIJ KLMNOPQRST UVWXYZ[*]^_
```

参照: USR, LOADM

APL CALL 1700

・機能は本文と同じ。

TRS_I —

CPM CALL A (A1, A2)

・先頭の変数は、開始アドレスを示す。ただし、この変数は配列であってはならない。
 ・引数の受け渡しができる。

M223 CALL #1, A1, A2, B1\$

・機械語プログラムに、0~31の番号をつけ、この番号で機械語プログラムを実行する。
 ・引数は、一つのプログラム文に8個まで指定できる。

MZK —

CBM SYS A

・機能は本文と同じ。

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・CPMと同じ。ただし、引数はすべての型の変数が利用できる。

レベル₃ ・本文参照。

C180 —

M243 ・M223と同じ。

MZB —

BUB —

FM8 ・本文と同じ。

PSP ・CPMと同じ。ただし、開始番地は、16進数の数値で指定する。
 ・開始番地と引数は、配列変数であってはならない。

PC88 ・CPMと同じ。ただし、引数はすべての型の変数を指定できるが、定数や式は渡せない。

N52 —

PC6 ・アドレスは省略できない。

MLT ・CPMと同じ。また、カッコ内は繰り返しでもよい。
 ・開始アドレスは、DEF SEG文で指定したセグメント番地に対するオフセット値で与える。

HC ・本文と同じ。

FP11 GO &H7000

・先頭番地を省略したときは、PCレジスタ内の番地から実行する。
 CALL &H7000, 0, 1, 2, 3
 ・機械語ルーチン呼び出す。RET命令でもとる。
 ・指定は、アドレス、引数A, HL, DE, BCの順。
 ・機械語ルーチンに対して、レジスタを通じて引数を渡せる。

SMC ・CPMと同じ。ただし、引数は10個まで。

MZ35 —

X1Hu ・APLと同じ。

MB16 ・MLTと同じ。なお、引数はすべての型の変数を指定できるが、定数や式は渡せない。

IBM55 ・MB16と同じ。

機械語プログラムの実行 EXEC

エグゼキュート

<p>PC100 ・ CPM と同じ。ただし、 A はオフセット値である。</p>			
<p>MSX —</p>			

機械語のセーブとロード MSAVE, MLOAD

EM・セーブ, EM・ロード

形式: ① MSAVE "AFILE", &7000, 2048
 ② MLOAD "BFILE", &7000, 1024

省略可

機能: ①機械語ファイルを, ファイル名, 先頭番地, 大きさの順で指定して, ディスク上へ格納する.
 ②ディスク上の機械語ファイルをファイル名, 先頭番地, 大きさを指定して, 本体上のメモリへ読み込む.

特徴: 1 (a) "AFILE" の前へ (*) をつけて * "AFILE" とすると, 秘密機械語仕様となる.
 (b) 機械語サブルーチンなどをディスク上へ保存するのに便利である.
 2 MLOAD を実行する際, 先頭番地, 大きさを省略すると, MSAVE 命令で指定された先頭番地, 大きさが有効となる.

プログラム例: "AFILE" という名の機械語プログラム・ファイルをディスクからロードし, BASIC プログラムのサブルーチンとして実行する.

```
10 MLOAD "AFILE", &8000, 1024
20 USR &8000
30 END
```

参照:

APL ・ CLI で実行可.

TRS_I ・ CLI で実行可.

CPM ・ CLI で実行可.

M223 ・ DOS モードでデバッグを使えば同じことができる.

MZK —

CBM —

PC8 —

TRS_{II} ・ CLI で実行可.

PC3 ・ 本文参照.

IF8₂ BSAVE "2: AFILE",
 &H8000, &H81FF, &H8010
 BLOAD "2: AFILE", &H2000
 ・ 機能は本文と同じ.

レブル₃ SAVEM "CAS1: AFILE",
 &H8000, &H8FFF, &H8000
 LOADM "CAS1: AFILE",
 &H8000
 ・ 機能は本文と同じ.

C180 —

M243 ・ M223 と同じ.

MZB —

BUB —

FM8 SAVEM "1: AFILE",
 &H8000, &H8FFF, &H8000
 LOADM "2: BFILE", &H2000,
 R
 ・ 機能は本文と同じ. &H2000はオフセット値.

PSP BSAVE "FD1",
 "AFILE", 1024, 2047, 1024
 BLOAD "FD2", "AFILE"
 ・ 機能は本文と同じ.

PC88 BSAVE "1: AFILE",
 &HE000, &H02FF
 BLOAD "2: BFILE", &H2000,
 R
 ・ 機能は本文と同じ. &H2000はオフセット値.

N52 —

PC6 BLOAD "1: AFILE",
 &HD00
 BSAVE "2: BFILE", &HD00,
 &H200
 ・ 機能は本文と同じ. ただし, 特徴の1 (a) と2はない.

MLT BSAVE "AFILE", オフセット, レングス
 ・ オフセット値は, 直前の DEF SEG 文で与えられた番地にオフセットを加算して与えられる.
 ・ オフセット値は 0 ~ 65535.
 ・ レングスの値は 0 ~ 65535.
 BLOAD "BFILE", オフセット
 ・ オフセット (省略可) 指定をする場合, そのプログラムは再配置可能な性質をもっていなくてはならない.

HC LOADM "B: AFILE",
 &7000
 ・ &7000はオフセット値で, SAVE M で指定した先頭番地に加算され, その結果番地からロードする. 大きさ指定はない.
 ・ LOADM ではデバイス名 "COM:" (カセット) を指定できない.
 SAVEM "AFILE", ---, V
 ・ SAVEM も LOADM と同様にセーブできる. この時, マイクロ・

機械語のセーブとロード MSAVE, MLOAD

エム・セーブ, エム・ロード

<p>カセットにセーブされ、自動的に CRC チェックがされる。</p> <p>FP11 SAVEM "0: AFILE", &H8000, &H8100, &H8020 ・指定順はデバイス名, ファイル名, 先頭アドレス, 終了アドレス, ス タート・アドレス (省略可). LOADM "0: AFILE", R ・R を指定すると読み込み終了後, 即実行する。 ・読み込むファイルは, SAVEM 文 で出力されたバイナリ型式のもの に限る。</p> <p>SMC LINK "DSK: AFILE" ・指定デバイス上の機械語ファイル を, ファイル名を指定して, 本体 上のメモリへ読み込む。 ・SAVE, LOAD, CSAVE, CLOAD で BASIC 命令と同時に セーブ, ロードできる。</p>	<p>文で与えられた番地にオフセット を加算して与えられる。 ・オフセット値は 0~65535. ・レンジスの値は 1 ~ 65535.</p> <p>IBM55 ・MB16 と同じ。</p> <p>PC100 ・PC88 と同じ。</p> <p>MSX ・BSAVE "デバイス名; ファイル名", 先頭番地, 終了番 地, 実行開始番地 BLOAD "デバイス名: ファイル 名", R, オフセット ・実行開始番地が省略されたとき, 先頭番地を実行開始番地とする。 ・R を指定すると, ロードしたあと 実行開始番地より実行する。 ・オフセットを指定すると, 先頭番 地にオフセットの値だけ加えた番 地にロードされる。</p>		
<p>MZ35 ・本文と同じ。ただし, 特 徴 1 の(a)は使えない。</p>			
<p>X1Hu LOAD "TEST", &HA000, R ・R を指定すると, 読み込み終了後 実行する。 ・アドレスを省略すると, SAVEM で指定したアドレスにロードされ る。 SAVEM "TEST1", &H7800, &H9EFF, &H9010 ・パラメータは, 先頭アドレス, 終 了アドレス, スタート・アドレス。 ・スタート・アドレスを省略すると, 先頭アドレスが設定される。</p>			
<p>MB16 BSAVE "A: AFILE", オフセット, レンクス BLOAD "A: BFILE", オフセ ット ・オフセット値は直前の DEF SEG</p>			

機械語モードへの移行 MON

モニター

形式：MON

機能：機械語モードに制御を移す。

- 特徴：
- 1 BASIC 実行モードから、機械語実行モードに制御を移す。
 - 2 BASIC プログラムはこわれない。
 - 3 **(CTL)-[B]** で、BASIC 実行モードにもどすことができる。
 - 4 機械語レベルでの命令は下記のとおり。

Sxxxx	指定されたアドレスの内容表示、内容変更
Dxxxx, yyyy	スタート・アドレス、エンド・アドレス間のメモリ内容表示
Wxxxx, yyyy	スタート・アドレス、エンド・アドレス間のメモリのテープへの書き込み
L	テープからのデータの読み込み
LV	テープからの読み出しデータと、メモリ内容ペリファイ
Gxxxx	指定されたアドレスへジャンプ
TM	メモリのテスト

実行例：機械語実行モードにして、機械語プログラムをロードして BASIC モードにもどす。

```

MON
*
*
*
OK

```

参照：

APL ・ RESET C080 として、
(CTRL)-[B] を押す (アップル・ソフト・バージョン)。

TRS CMD "D"
・機能は、本文と同じ。また、SYSTEM とすると機械語をロードすることができる。

CPM ・該当はない。ただし、同じことをするには、BASIC モードからぬけて機械語モードにすればよい。

M223 ・該当はない。ただし、デバッグを呼べば特徴4などを実行できる。

MZK BYE
・機能は、本文と同じ。ただし、モニタ機能は少ない。

CBM SYS54386
・本文と同じ。ただし、特徴4は、M：メモリ表示 X：BASIC へのリターン R：レジスタ表示 L：16進データ・ロード G：指定アドレスへジャンプ S：16進データ・セーブ

PC8 ・本文参照。

TRS —

PC3 —

IF82 —

レベル3 ・本文と同じ。ただし、特徴4は、M：メモリ内容変更 G：指定アドレスにジャンプ R：レジスタ内容表示、変更 D：指定アドレスから64バイト表示 **(STOP)**、**(CTRL)-[C]**、**(CTRL)-[D]** など で BASIC リターンする。

C180 —

M243 ・ M223 と同じ。

MZB ・本文と同じ。ただし、BASIC へのリターンは (J) を入力する。

BUB —

FM8 ・本文と同じ。機械語モニタはレベル3と同じ。ただし、**(STOP)**、**(CTRL)-[C]**、**(CTRL)-[X]** キーなどで BASIC にもどる。

PSP ・本文と同じ。ただし、特徴4は、M：メモリ内容変更 J：指定アドレスにジャンプ R：レジスタ内容表示、変更 B：BASIC へのリターン。

PC88 ・本文と同じ。ただし、コマンドは約3倍にふえている。
(HELP) キーを押すとコマンドの説明がリストされる。

N52 —

PC6 —

MLT —

HC ・特徴4で、S、D、W は同じ。その他はない。また、ほかに種々の命令がある。
・特徴3は、**(B)** のみで抜け出せる。

FPI1 ・BASIC モードにもどるには、BA コマンドを実行する。
・特徴4は、DM：メモリ内容表示、CM：メモリ内容変更、MM：メモリ内容転送、GO：プログラム実行、CR：レジスタ内容変更、ST：テープヘセーブ、LT：テープからロード。

機械語モードへの移行 MON

モニター

<p>SMC ・該当はない、ただし、 SYSTEM で DOS モードにして、 同様なことができる。</p>			
<p>MZ35 BYE ・FDOS 中に DEBUG コマンドが あるので、これを用いて機械語の デバックが行える。</p>			
<p>X1Hu ・本文と同じ、BASIC へ のリターンは R。 ・P：プリンタ・スイッチ、D：ダ ンプ、M：メモリ・セット、F： ファインド、G：実行、T：転送、 S：カセットへセーブ、L：カセ ットからロード、V：ベリファイ。</p>			
<p>MB16 ・該当はない、ただし、 MS-DOS 中に DEBUG コマンド があるので、これを用いて機械語 のデバックが行える。</p>			
<p>IBM55 ・MB16 と同じ。</p>			
<p>PC100 ・SMC と同じ。</p>			
<p>MSX —</p>			

インプット

キーボードからの入力 INPUT

形式: INPUT "タテ"; A, E\$

省略可 *省略・繰り返し可
 プrompt文 リスト

機能: キーボードから変数に入力する。

- 特徴: 1 LINE INPUT 文とは異なり、各データの項目は (,)、スペースで区切り、また入力時に (?) が出る。
 2 入力されるデータ数は、リスト中の変数名の数と一致していなければならない。
 3 Prompt文がある場合、Prompt文のあとに (?) が出る。
 4 リストにあげる変数名は、数値変数名でも文字変数名(添字つきを含む)でもかまわないが、入力されるデータの型に一致していなければならない。

プログラム例: 長方形の面積を計算する。

```
10 INPUT "タテ "; A1
20 INPUT "ヨコ "; A2
30 PRINT "面積は="; A1*A2
40 END
```

実行例

```
タテ ? 5
ヨコ ? 5
面積は = 25
```

参照: LINE INPUT, INPUT #n

APL ・本文と同じ。ただし、Prompt文を付けた場合、(?)は出ない。

TRS_I ・各項目は (,) で区切る。
 ・スペース、コロン、コンマを含む文字ストリングの場合は、ストリングを (") で囲む。

CPM ・本文と同じ。

M223 INPUT "タテ", A, E\$
 ・Prompt文は省略可。
 ・Prompt文を省略した場合、疑問符 (?) は出るが、Prompt文を付けると疑問符は出ない。

MZK ・(?) は出ない。

CBM ・入力できるデータ長は最大80文字。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 INPUT "タテ": A, B;
 "ヨコ", C
 ・Prompt文は省略可。
 ・区切りにセミコロンを用いると、表示間隔が詰み、コンマを用いると1領域(20文字)単位に表示される。

IF8₂ ・INPUT: "タテ": A または、INPUT "タテ", A でも可。
 ・Prompt文は省略可。
 ・Prompt文があつて、その後の区切りがコンマである場合、(?) は出力されない。

レベル3 INPUT "タテ"; A または INPUT "タテ", A
 ・Prompt文は省略可。
 ・変数名は繰り返し可。

・(,) や (:) も引用符 (") で囲むことにより入力可。

C180 INPUT "タテ": E\$
 ・Prompt文は省略可。
 ・その他は M223 と同じ。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・各項目は (,) で区切る。

FM8 ・INPUT "タテ", A, E\$
 でも可。
 ・Prompt文は省略可。
 ・Prompt文の後がコンマの場合、(?) は出ない。

PSP ・Prompt文を省略した場合、(?) は出ない。
 ・TAB, WID 関数も使用可。
 ・255文字まで入力可。

PC88 ・IF8₂ と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・文字定数の入力では (") で囲む必要はない。

FP11 INPUT "タテ", A でも可。このとき (?) は出ない。
 ・各項目はコンマ (,) で区切る。
 ・文字列の中にコンマ (,) を含まない場合両端の (") は省略できる。

SMC ・IF8₂ と同じ。ただし、INPUT: "タテ": A は不可。

キーボードからの入力

INPUT

インプット

<p>MZ35 ・ MZK と同じ。また、KEY、KEYIN でも形式を指定して同様のことができる。</p>			
<p>X1Hu ・ 本文と同じ。なお、リターン・キーだけだと前の値を保持する。</p>			
<p>MB16 ・ IF8₂ と同じ。またセミicolon (;) が INPUT のすぐ後に続いた場合、カーソルはデータ入力行に残る。</p>			
<p>IBM55 ・ MB16 と同じ。</p>			
<p>PC100 ・ MB16 と同じ。</p>			
<p>MSX ・ 本文と同じ。</p>			

行入力文 LINE INPUT

ライン・インプット

形式: LINE INPUT "タテ"; E\$

省略可
プロンプト文

機能: 1 行分をそのまま一つの文字変数に入力する。

- 特徴: 1 INPUT 文と異なり、コンマ (,), 前後の空白、引用符 (') などとも入力できる。
 2 改行 (RETURN) で 1 行の区切りとなる。
 3 1 行は最大 255 文字。
 4 入力時に (?) は画面に出ない。
 5 スクリーン・エディタ機能もあり、入力時にカーソルを、画面上に表示されている文字列の行に移し、(RETURN) キーを押すことによってその行をそのまま入力できる。
 6 データの誤りチェックをしながら入力するとき、またはデータ中のコンマの有無が入力時に不明のときなどに使うと便利である。

用語: モニタ プログラムの実行および制御、またはプログラムの実行にともなう周辺機器の動作の制御、および監視を行うために用意されたプログラム。

プログラム例: "list, 10", または "オワリ" のような文字列を E\$ に入力する。

```
10 LINE INPUT "コメント" ?"; E$
20 IF E$="オワリ" THEN END
30 PRINT E$
40 GOTO 10
```

実行例

```
コメント ?list,10
list,10
コメント ?オワリ
```

参照: LINE INPUT #n, INPUT

APL —

TRS: ・本文と同じ。

CPM LINE INPUT "タテ"; E\$
 ・プロンプト文、始めの; は省略可。
 ・1 行は最大 254 文字。

M223 INPUT LINE "タテ"; E\$
 ・プロンプト文は省略可。
 ・最大入力文字数は 255。
 ・E\$ は DIM 宣言しなければ 10 文字までしか入力できない。
 ・特徴 5 はない。

MZK ・該当なし。ただし、モニタリ、サブルーチン GETL で同様のことが可能。

CBM —

PC8 ・本文参照。
 ・実行中 (STOP) キーを押して中断し、また (CONT) キーで再開すると、再度 1 行分の先頭からの入力になる。

TRS: ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。ただし、LINE INPUT "タテ"; E\$; も可。このとき改行はしない。
 ・最大入力文字数は 254。

レベル₃ ・本文と同じ。ただし、LINE INPUT "タテ"; E\$ も可。

C180 LINPUT "タテ"; E\$
 ・プロンプト文は省略可。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・レベル₃ と同じ。

PSP LINPUT "タテ"; TAB (10); E\$
 ・プロンプト文、TAB、WID 関数は省略可。
 ・最大入力文字数は 255。ただし、WID 関数で入力可能文字数を指定できる。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・最大 254 文字。

HC ・本文と同じ。

FP11 LINE INPUT "タテ"; E\$, n
 ・n は文字定数。省略可。省略すると 255。

SMC INPUT/L "タテ"; E\$
 ・(;) の代わりに (,) を使用すると疑問符を表示しない。

MZ35 —

X1Hu ・本文と同じ。なお、(;) の代わりに (,) でもよい。
 ・LINPUT も使える。

MB16 ・本文と同じ。ただし、LINE "タテ"; E\$; も使える。このときは改行しない。

IBM55 ・CPM と同じ。ただし、INPUT 直後にセミコロンがある

キー入力の検出 INKEY\$

イン・キー\$

形式: INKEY\$

機能: キーボードからの入力を1文字瞬間的に検出する。

- 特徴: 1 キーが押されるとその文字が得られ、キーが押されないときは、ナル・ストリング¹⁾を得る関数である。
 2 キーを押しても画面には表示されない。
 3 (STOP) キー, (CTRL) キーは検出できない。
 4 通常は文字変数に代入して使う (例 E\$=INKEY\$)。

用語: ¹⁾ナル・ストリング 長さが0の文字列。""で表現する。未定義の文字列はこの状態になっている。

プログラム例: "E"が入力されるまで、プリンタにキーボードからの入力を印字する。

```
10 A$=INKEY$:IF A$="" THEN 10
20 LPRINT "KEY INPUT= ";A$
30 IF A$="E" THEN LPRINT"END":END
40 GOTO 10
```

実行例

```
KEY INPUT= L
KEY INPUT= O
KEY INPUT= V
KEY INPUT= e
KEY INPUT= E
END
```

参照: INPUT\$

APL GET E\$

- ・引数は数値変数, 文字変数とも可。
- ・数値変数のとき, 0~9までの数字キーのみを受け付け, ほかのキーを押すとエラーになる。
- ・実行は1文字入力するまで次のステートメントには移らない。

TRS₁ ・本文と同じ。

CPM —

M223 —

MZK ・APLと同じ。

CBM ・APLと同じ。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 KEY E\$

- ・キーボードから1文字E\$に入力する。
- ・E\$を省略すると以前の入力キーをクリアする。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 —

M243 —

MZB ・APLと同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP KIN\$

- ・機能は本文と同じ。キー入力がないとき NULL フラグが立つ。

NULL 関数で読み取れる。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FPI1 ・本文と同じ。

SMC INKEY\$ (N)

- ・N=0の時実行直前までに押されたキーの最後のデータがなければナル・ストリングを返す。
- ・N=1の時 INKEY\$ 実行後最初に押されたキーのデータを返す。
- ・N=2の時本文と同じ。
- ・Nの指定を省略すると0を指定したことになる。

MZ35 ・PC3と同じ。

X1Hu INKEY\$ (N)

- ・N省略のときキーが押されたらその文字, 押されていないければナル・ストリング。
- ・N=0のときキーが押されたら間断なくその文字, 押されていないければナル・ストリング。
- ・N=1のときカーソルをブリンクし1文字押されるまで待つ。
- ・N=2のときキーボードからの入力状態を示す。SUBCPUからの1バイトの情報を与える。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。ただし、漢字モード中に押されたキーと、BREAK, PAUSE, COPY, リセットは検出できない。

キー入力の検出 INKEY\$

イン・キー\$

<p>PC100 ・ (CTL) - (C) と (STOP) は検出できない。入力対象はキー ボードでなく、キーボード・バッ ファである。</p>			
<p>MSX ・ 本文と同じ。</p>			

一定文字数の入力 INPUT\$

インプット\$

形式: INPUT\$ (M1, #M2)

省略可

省略可

機能: 番号M2のファイル, またはキーボード (M2省略時) からM1個の文字列を入力する関数。

- 特徴: 1 キーボード入力の場合には, 入力した文字は CRT に表示されない。
- 2 この関数を中断するキー ((CTL)-[C] または (STOP)) 以外はそのまゝ入力される。
- 3 M1の範囲は, $0 \leq M1 \leq 255$ で, 小数点以下は切り捨てられる。
- 4 M1個の文字列が入力されると, (CR) キーを押さなくても入力が終了する。
- 5 INPUT 文や LINE INPUT 文では, 入力することのできない改行文字なども入力が可能であるので便利である。

プログラム例: INP 関数と INPUT\$ 関数を組み合わせた例。入力文字の最後が@ならば終了する。

```

10 LPRINT "IF LAST TYPE 'Q' KEY THEN END"
20 FOR I=4 TO 10 STEP 2
30 IF INP(2)=254 THEN END
40 A$=INPUT$(I)
50 LPRINT A$
60 NEXT I

```

実行例

```

IF LAST TYPE 'Q' KEY THEN END
TAKE
CHAN!Q

```

参照: INPUT, LINE INPUT

APL GET A\$

TRS₁ —

CPM ・本文と同じ。

M223 ・該当なし。ただし、キーボードの1文字のみの入力は、INPUT #0, E\$。

MZK ・該当なし。ただし、INP 参照。

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 —

IF₈₂ ・本文と同じ。レベル₃ ・本文と同じ。ただし、ファイル番号が0のときも、キーボード入力である。

C180 —

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・(CTL)-[C] でも入力される。

PSP —

PC88 ・本文と同じ。

N52 ・ファイルはシーケンシャルでオープンされていなければならない。

・M1は1から255まで。

PC6 —

MLT ・本文と同じ。

HC ・特徴2, 3, 4は不可。

FP11 ・(BREAK) キーと (STOP) キーを除くすべてのキーの入力ができる。

・(BREAK) キーで中断するが、(STOP) キーでは中断しない。

SMC —

MZ35 —

X1Hu ・本文と同じ。

・特徴2はBREAKコード以外は入力できる。

MB16 ・本文と同じ。ただし、(STOP) キーはなく (CANCEL) キーである。

IBM55 ・M1バイトの文字列を入力する。

・M1の小数点以下は四捨五入。

・漢字まじり文のときは INPUT ¥ を用いる。

PC100 ・IBM55 と同じ。

MSX ・本文と同じ。

シーケンシャル・ファイル入力文 INPUT #n

インプット #n

形式: INPUT #5, A, C#, E\$

*省略・繰り返し可

機能: シーケンシャル・ファイル¹⁾からデータを入力する。

- 特徴: 1 ファイル番号は、OPEN 命令ですでにオープンされたものでなければならぬ。
 2 入力時に (?) は画面に出ない。
 3 ファイル中のデータは、変数と型が一致しなければならない。
 4 (,) は INPUT 命令と同じ。
 5 PRINT #n で書かれた内容を読む時は INPUT #n を使う。

用語: ¹⁾シーケンシャル・ファイル ファイル内のデータが一つずつ順番に書き込まれ、同じように順番に読み出されるもの。シーケンシャル・ファイルは可変長レコードより成り立っている。☐ランダム・ファイル。

プログラム例: ファイル "AFILE" を作り、書き込み、読み出す。

```
10 PRINT "make & open AFILE": OPEN "AFILE" FOR OUTPUT AS #5
20 PRINT #5, "TEST PROGRAM"
30 PRINT "close AFILE": CLOSE 5
40 PRINT "open AFILE": OPEN "AFILE" FOR INPUT AS #6
50 INPUT #6, E$
60 PRINT E$
70 PRINT "close AFILE": CLOSE 6
80 END
```

実行例

```
make & open AFILE
close AFILE
open AFILE
TEST PROGRAM
close AFILE
```

参照: INPUT, OPEN, PRINT #n

APL D\$ = CHR\$ (4) :
PRINT D\$: "READ AFILE" :
INPUT A, E\$

TRS_I ・本文と同じ。

CPM ・本文と同じ。ただし、
カセット・レコーダは扱えない。

M223 ・本文と同じ。ただし、
カセット・レコーダからの入力は
OPEN 命令でカセット・ファイル
をオープンして用いる。
・入力時に疑問符が画面に出る。

MZK ・CPM と同じ。

CBM ・本文と同じ。ただし、
カセット・レコーダからの入力は
OPEN 命令でカセット・ファイル
をオープンして用いる。

PC8 ・本文参照。
・ファイル番号を -1 にするとカセ
ット・レコーダから入力する。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。ただし、
カセット・レコーダからの入力は
CLOAD 命令を用いる。

IF8₂ ・CBM と同じ。レベル₃ ・CBM と同じ。

C180 ・可変長の場合 INPUT #1 :
A,
・固定長の場合 GET #1 : A

M243 ・M223 と同じ。

MZB ・本文と同じ。ただし、
カセット・レコーダからの入力は
INPUT/T.

BUB ・本文と同じ。

FM8 ・CBM と同じ。

PSP GET #1, A, E\$ EOF
1000
・ファイルが終わると行番号1000へ
飛ぶ。

PC88 ・本文と同じ。

N52 ・変数と型が一致しない場
合は 0 が入る。
・ファイル中のデータは、コンマか
CHR\$ (13) と CHR\$ (10) で
区切る。
・数値の場合は空白も区切りになる。
・もしデータの最初が (") の場合
は次の (") ままで読み込まれ、
(") で囲まれた文字列は (")
を文字として含むことはできない。
・255 文字の文字データが読み込ま
れたときも区切られる。

PC6 ・INPUT#-0: キーボード。
・INPUT#-1: カセット・テープ。
・INPUT#-2: RS-232C.

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・ファイルは OPEN/I で開
かれている必要がある。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。

MB16 ・N52 と同じ。

IBM55 ・本文と同じ。

シーケンシャル・ファイル入力文 INPUT #n

インプット #n

<p>PC100 ・シーケンシャル・ファイルはディスクに限らず RS-232C ポート、キーボードも対象となる。</p>			
<p>MSX ・本文と同じ。</p>			

シーケンシャル・ファイル行入力文 LINE INPUT #n

ライン・インプット #n

形式：LINE INPUT #5, E\$

機能：シーケンシャル・ファイルより、1行文を文字変数に入力。

- 特徴：1 ファイル番号は、OPEN 命令ですでにオープンされたものでなければならない。
- 2 プロンプト文が使えない点と、ファイルよりの入力である点を除き、そのほかは LINE INPUT 命令と同じ。

プログラム例：(,)を含んだファイル"AFILE"を読み込む。

```
10 OPEN "AFILE" FOR INPUT AS #5
20 IF EOF(5) THEN GOTO 50
30 LINE INPUT #5,A$:PRINT A$
40 GOTO 20
50 PRINT"End": CLOSE 5
60 END
```

実行例：

```
Test file, 1'st line.
Test file, 2'nd line.
Test file, 3'th line.
Test file, 4'th line.
Test file, 5'th line.
Test file, last line.
End
```

参照：LINE INPUT, OPEN

APL —

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 INPUT LINE #5, E\$
・機能は本文と同じ。

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。C180 LINPUT #5: E\$
・機能は本文と同じ。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・CHR\$(13)に続くCHR\$(10)コードまでのすべての文字を区切ることなく、指定された文字変数に読み込むこともできる(ただし、読み込める文字数の最大は255まで)。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC INPUT/L #S
・機能は本文と同じ。ただし、ファイルは OPEN/I で開かれているもの。

MZ35 —

X1Hu ・本文と同じ。なお、最大入力文字数 255 文字。
・LINPUT# も同じ動作。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

ランダム・ファイルからの入力 GET

形式: GET #9, 3

□ □

省略可

機能: ランダム・ファイル¹⁾の指定された1レコード²⁾を読み込む。

- 特徴: 1 この命令を実行する前に、OPEN 文、FIELD 文を実行しなければならない。
- 2 読み込まれたデータは、FIELD 文で指定した変数に入る。
- 3 3はレコード番号。これを省略すると最後の GET 文、または PUT 文の次のレコードが引き続いて指定される。
- 4 ランダム・ファイルは、データ・ファイルの途中から引き出す必要のあるデータの格納に便利であり、そのデータを読むにはこの命令が必要である。

用語: ¹⁾ランダム・ファイル ファイル内のデータの書き込み順序とは関係なく、任意のデータを読み出したり、任意の場所へ書き込んだりできるファイルのこと。固定長レコードより成り立っている。

²⁾レコード 1回に入力される組のこと。物理的な入出力（セクタ単位）ではなく論理的な入出力である。

プログラム例: ランダム・ファイル "AFILE" の作成、書き込み、読み出し。

```
10 OPEN "AFILE" AS #9
20 FIELD #9,20 AS E$
30 FOR I=1 TO 5
40 LSET E$="test data"+STR$(I)
50 PUT #9,I
60 NEXT I
70 FOR I=5 TO 1 STEP -1
80 GET #9,I
90 PRINT E$;
100 NEXT I
110 CLOSE 9
120 END
```

実行例

```
test data 5    test data 4    test data 3
test data 2    test data 1
```

参照: OPEN, PUT, FIELD, INPUT #n

```
APL D$=CHR$(4):PRINT
D$;"READ AFILE, R";3:
INPUT A, E$
```

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 GET #1 RECORD 3

MZK INPUT #1 (3), A, E\$

CBM ・RECORD 命令でポインタを設定した後、INPUT #1, A, E\$。

PC8 ・本文参照。

TRS₁₁ GET 1, 25
・1はバッファ番号、25はレコード番号。

PC3 ・本文と同じ。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 READ #1:3, E\$

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC GET% 0, A/ (RUN ファイルのデータ読み込み)
・0はレコード番号。
・A/は変数名。
・GET%実行以前に DEFFILE 文により、1レコード長さ、そのレコード長を含むファイルの位置を定義しておく。

FP11 ・本文と同じ。

SMC ・FIELD 文の代わりに RECORD 文を使用する。

MZ35 ・本文と同じ。なお、本文の FIELD 文は RFORMAT を使う。

X1Hu ・本文と同じ。

MB16 ・ファイルがコミュニケーション・ファイルの場合、3は読み込むデータのバイト数。
・レコード番号の入力数は (EOF/レコード長) あるいは 32767 のいずれか小さいほうまで。

IBM55 ・本文と同じ。ただし、レコード番号は1から32767の範囲。

PC100 ・本文と同じ。

MSX —

内部コードのファイルからの入力 READ #n

リード #n

形式：READ #5, A, C#, E\$

*省略・繰り返し可

機能：シーケンシャル・ファイルより、内部コードを変数に入力する。

- 特徴：
- 1 ファイル番号は、OPEN 命令ですでにオープンされたものでなければならぬ。
 - 2 WRITE #n で書かれたファイルは、READ #n で読まなければならない。
 - 3 ファイル番号は 1 ～ 8。

プログラム例：内部コードでファイルに書き込み、読み出す。

```

10      OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20      OPEN "AFILE" FOR OUTPUT AS FILE 6
30      FOR I = 1 TO 5
40          WRITE #6 , I
50      NEXT I
60      CLOSE 6
70      OPEN "AFILE" FOR INPUT AS FILE 5
80      FOR I = 1 TO 5
90          READ #5 , A
100         PRINT #1 , "DATA=" ; A
110      NEXT I
120      CLOSE 5
130      CLOSE 1
140      END
    
```

実行例

```

DATA= 1
DATA= 2
DATA= 3
DATA= 4
DATA= 5
    
```

参照：OPEN, WRITE #n, INPUT #n

APL —

TRS₁ —

CPM —

M223 ・本文参照。

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 BINPUT #1, A, E\$

IF8₂ —

レベル₃ —

C180 —

M243 ・本文と同じ。

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC ・INPUT #n 参照。

FP11 —

SMC RECEIVE #5, A, E#

- ・ファイルは OPEN/R ですでにオープンされたものでなければならぬ。
- ・SEND #n で書かれたファイルは RECEIVE #n で読まなければならぬ。
- ・ファイル番号は 0 から 127。

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 ・INPUT #n 参照。

MSX —

ポートからの入力 INP

インポート

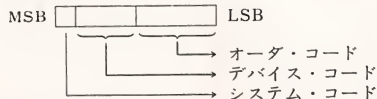
形式: INP (M)

機能: 引数¹⁾のポート²⁾のレジスタ³⁾の内容(1バイト⁴⁾)を得る関数。

特徴: 1 0 ≤ M < 255.

2 複数のキーが同時に押されていても、プログラムで解析できる。
また、外部の機器のデータをポートから入力するときに使う。

3 (a)ポート番号(I/O アドレス8ビット)は、次のように決められている。



(b)システム・コード: CPU 基板にあるポートのアドレスと、拡張インターフェースのポートのアドレスとを定義する。0: 基板, 1: 拡張

(c)デバイス・コード: 機器の種類を示すコード。

000: キーボード
001: プリント・データ
010: シリアル・チャネル
011: CPU システム・コントロール
100: ストロープ・ポート
101: CRTC コントロール
110: DMAC コントロール
111: 未使用

(d)オーダ・コード: 各デバイス・コードで選択された機器内部の
コマンドを指定する。例えば、INP (0)~INP (9) は、キーボード、INP (16) は、
プリンタの出力データ。INP (100), INP (101), INP (104) は、DMA コントローラ。
INP (80), INP (81) は、CRT コントローラ。

(e)キーボードの場合のオーダ・コードは、次のとおり。

00... 0 ~ 7 (テン・キー)
01... 8, 9, *, +, =, ,, ., RET (テン・キー)
02... @, A ~ G
03... H ~ O
04... P ~ W
05... X ~ Z, [, \,], △, □
06... 0 ~ 7
07... 8, 9, :, ;, ', /, △
08... HOME, ↓, ←, INS, GRAPH, カナ,
SHIFT, CTRL
09... STOP, f.1, f.2, f.3, f.4, f.5, SPACE
ESC

プログラム例: INP 関数と INPUT\$ 関数を組み合わせた例。入力文
字の最後が@ ならば終了する。INP (2) はキーボード。

```
10 LPRINT "IF LAST TYPE '0'
    KEY THEN END"
20 FOR I=4 TO 10 STEP 2
30 IF INP(2)=254 THEN END
40 A$=INPUT$(I)
50 LPRINT A$
60 NEXT I
```

実行例

```
IF LAST TYPE '0' KEY THEN END
TAKE
CHAN!0
```

参照: OUT

用語: ¹⁾引数 関数およびサブルーチンを引用する時のパラメータ。この場合Mを指す。
²⁾ポート コンピュータと外部機器との間で、データのやりとりを行うときの、デ
ータの出入口。ポートには、識別のため番号が付けられている(ポート番号)。
³⁾レジスタ コンピュータが仕事をする時メモリのほかにデータを一時的に記憶す
る部分。
⁴⁾バイト 8ビットをまとめて1バイトと定義する。1 K(キロ)バイトは1024バイ
ト。

ポートからの入力 INP

インポート

APL ・該当なし。ただし、コマンドで IN#(M) が似た命令である。 ・ゲーム・パドルの入力は PDL で行う。	PC88 ・本文と同じ。	
TRS_I ・本文と同じ。	N52 —	
CPM ・本文と同じ。	PC6 ・タッチ・パネルの入力は PAD で行う。 ・ジョイスティックの入力は STICK で行う。	
M223 ・INP (214) がキーボード。	MLT ・ $0 \leq M \leq 65535$ 。	
MZK INP @M, A ・I/O ポート番号 M にあるデータを変数 A に読み出す。	HC —	
CBM ・該当なし。 ・入出力操作直後のコンピュータの状態は ST で与えられる。	FP11 ・ $-32769 < M < 65536$ 。	
PC8 ・本文参照。	SMC ・ポートの割り付けについては異なるが機能は本文と同じ。	
TRS_{II} —	MZ35 —	
PC3 —	X1Hu ・I/O ポートから 1 バイトのデータを読み込む。 ・引数は 0 ~ &HFFFF。	
IF8₂ ・ $-32768 \leq M \leq 32767$ 。	MB16 ・本文と同じ。ただし、 $0 \leq M \leq 65535$ 。	
レベル₃ —	IBM55 ・ SMC と同じ。ただし、 $0 \leq M \leq 65535$ 。	
C180 ・通信回線をファイルとみて INPUT# 文で入力できる。	PC100 ・本文と同じ。	
M243 ・ M223 と同じ。	MSX ・本文と同じ。	
MZB ・ MZK と同じ。		
BUB —		
FM8 ・該当なし。ただし、アナログ・ポート入力関数として、ANPORT がある。		
PSP IN # (M) ・機能は本文と同じ。		

CRT への出力 PRINT

プリント

形式: PRINT A, "タテ"; B

省略可 * 省略・繰り返し可

リスト

機能: CRT 画面上に情報を出力する。

- 特徴: 1 リストを省略した時は、改行だけが行われる。
 2 1行を14文字ずつの二つまたは五つに分けて、それぞれを領域と呼ぶ。もし、式のリスト内でコンマ(,)で区切ると、次の値はその次の領域の始めから印字される。もしセミコロン(;)を使うと、次の値は最後の値の次のカラムから印字される。
 3 もし、式のリストの最後がコンマ(,)またはセミコロン(;)で終わっている場合、次の PRINT 文による印字は引き続いて特徴2のルールに従って始まる。コンマ(,)でもセミコロン(;)でも終わっていない場合は改行される。
 4 (?)が代わりに使える機種がある(シンボルの項参照)。
 5 空白を(;)の代わりに使用可。

プログラム例: コンマ(,)とセミコロン(;)による出力の違いを示す。

```
10 A=10
20 PRINT A;-A
30 PRINT A:A
40 END
```

実行例

```
10          -10
10 -10
```

参照: LPRINT, LPRINT #n, PRINT/L

APL ・ 本文と同じであるが、コンマで区切ると16文字単位の三つの領域に分けて出力。
 ・ コンマでもセミコロンでも区切らない場合は続けて出力する。

TRS_I ・ 本文と同じであるが、コンマで区切ると16文字単位の四つの領域に分けて出力。

CPM ・ 本文と同じ。

M223 ・ 本文と同じであるが、コンマで区切ると15文字単位の領域に分けて出力。
 ・ ダイレクト・コマンドとして実行するには TYPE M1 を用いる。

MZK ・ 本文と同じであるが、コンマで区切ると10文字単位の領域に分けて出力。

CBM ・ TRS_{II} と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じであるが、コンマで区切ると8文字単位の10の領域に分けて出力。

PC3 DISP A, B; "タテ"
 ・ 機能は本文と同じであるが、コンマで区切ると20文字単位の領域に分けて出力。

IF8₂ ・ 本文と同じ。

レベル₃ ・ 本文と同じであるが、変数と引用符で囲まれた文字列の区切りに限り、空白で区切ることが可能。この場合はセミコロンと同様である。

C180 ・ 本文と同じであるが、TAB, CSR 関数も使用可。

・ 領域は出力項目の式の型により異なる。

M243 ・ M223 と同じ。

MZB ・ MZK と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じであるが、TAB, WID 関数も使用可。
 ・ コンマで区切ると10文字単位の領域に分けて出力。これは WID 関数で変更可。
 ・ WRITE 文を PRINT 文の代わりに使用可。

PC88 ・ 本文と同じ。

N52 ・ 変数と文字定数、文字定数と文字定数との区切りに限り区切り記号を省略しても(;)を指定したときと同等の効果が得られる。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ 出力要素間の区切りはスペースを使用できる。また、出力要素同士が明確に区別できる場合区切りは省略できる。この場合いずれも(;)と同じ動作をする。

SMC ・ コンマで区切ると8桁のタブとして働き、データが8桁未満のとき次のタブ9桁目から表示し、8桁以上だと8桁タブセットされる。
 ・ さらに、PRINT (10, 10); "A", (11, 11); 321 は LOCATE 文を

CRT への出力 PRINT

プリント

兼ねたコマンドである。			
MZ35 ・ PC3 と同じ。			
X1Hu ・ MZK と同じ。 WRITE A,B ・ 画面に式の値を (,) で区切り詰めて表示。文字型は (") で囲んで表示する。			
MB16 ・ 本文と同じ。なお、 WRITE 文で代用できる。			
IBM55 ・ 本文と同じ。ただし、変数を空白で区切ることもできる。この場合は、セミコロンと同様。 ・ WRITE 文で代用ができる。			
PC100 ・ MB16 と同じ。			
MSX ・ 本文と同じ。			

書式指定プリント (その1) PRINT USING

プリント・ユージング

形式: PRINT USING " ! タテ & & ";

E1\$, E2\$

フォーマット文字列

*省略・繰り返し可

機能: 数値や文字列を, フォーマット文字列で指定した書式で CRT に出力する。

- 特徴: 1 文字列のフォーマット (フォーマット文字列: 機能).
- (1) ! : 与えられた文字列の, 最初の 1 文字だけをプリントする。
- (2) &_& : 与えられた文字列の, 最初の n + 2 文字を左詰めでプリントする (n = 0, 1, 2, ...).
- 2 フォーマット文字列中の (&), (!), (#), (.), (.), (+), (-), (Y), (*), (^) 以外の文字は, そのまま対応する位置に印字される。
- 3 フォーマット文字列中の指定の数が, 出力する変数の数より少なかったときは, 同じフォーマット文字列が繰り返し適用される。
- 4 前もって長さのわからない文字列を, 定められた位置に左詰めで出力するとき便利である。

プログラム例: 行番号 20 では, 文字列の最初の 1 文字を出力 (3 の機能が働いている) し, 30 行では, 最初の文字列の 2 文字, 次の文字列の 4 文字を出力する。また, フォーマット文字列中の (=) と (CM) はそのまま出力する。

```
10 E$="XY"
20 PRINT USING "!"$E$, "ABCDEF6"
30 PRINT USING "&&=& &CM "$E$;"1234"
40 END
```

実行例

```
XA
XY=1234CM
```

参照: PRINT, PRINT #n USING, LPRINT USING

APL —

TRS₁ : (1) は本文と同じ。
 (2) は, %_% とすれば同じことができる。

CPM : (1) は本文と同じ。
 (2) は, /_/_ とすれば同じことができる。

M223 : 文字列のフォーマットは本文と異なる。
 (1) << : < で指定した数だけ文字列を, 左詰めでプリントする。
 (2) >> : > で指定した数だけ文字列を, 右詰めでプリントする。
 出力先は, ライン・プリンタでもディスクでも同じ形式。

MZK —

CBM —

PC8 : 本文参照。

TRS_n : (1) は本文と同じ。
 (2) は CPM と同じ。

PC3 : PRINT USING は, プリント出力となる。CRT 出力は, DISP USING である。
 IMAGE 文によって, フォーマット文字列を指定することもできる。
 100 DISP USING 170: E\$
 :
 170 IMAGE "3A" は,
 100 DISP USING "3A": E\$
 と同じ働きをする。
 文字列のフォーマットは本文と異なる。AA または nA で文字の桁を指定し, 左詰めでプリントする。

IF8₂ : (1), (2) は本文と同じ。レベル₃ : IF8₂ と同じ。

C180 : フォーマット文字列の後にはコロン (:)。

IMAGE 文によって, フォーマット文字列を指定することができる。
 フォーマット文字列中に複数の書式を指定する場合はセミコロン (;) で区切る。
 100 PRINT USING 170: E\$
 : IMAGE 文の行番号
 170 IMAGE: <XX

書式文字列(" ")を使わない
 IMAGE 文で書式文字列に文字列変数は使えない。
 文字列のフォーマットは, 本文と異なる。<XX: X で指定した数だけ文字列を, 左詰めでプリントする。>XX: X で指定した数だけ文字列を, 右詰めでプリントする。
 特徴 3 はない。

M243 : M223 と同じ。

MZB —

BUB : 本文と同じ。

FM8 : 本文と同じ。

PSP : (1) は本文と同じ。
 (2) は TRS₁ と同じ。

PC88 : (1), (2) は本文と同じ。
 本文にない機能として, @ : 文字列を @ に代入して出力できる。
 PRINT USING "This is a @.":
 "book"
 ∴ This is a book.

N52 : 本文と同じ。

PC6 —

MLT : 本文と同じ。

HC : 本文と同じ。

書式指定プリント（その1） PRINT USING

プリント・ユーザング

FPI1 ・ PC88 と同じ。			
SMC ・ 特徴1の(2)は「」 で同じことができる。 ・ & は文字型データをそのまま表示する。			
MZ35 ・ PC3 と同じ。			
X1Hu ・ 本文と同じ。			
MB16 ・ PC88 と同じ。			
IBM55 ・ 1 (1) (2) のほか、 可変長ストリング欄指定の (@) がある。			
PC100 ・ PC88 と同じ。			
MSX ・ PC88 と同じ。			

書式指定プリント (その2) PRINT USING

プリント・ユージング

形式: PRINT USING "###.##"; A, A1

フォーマット文字列 *省略・
繰り返し可

機能: 数値を指定の位置に出力する。小数点の位置を指定することもできる。

特徴: 1 フォーマット文字列中の、一連の(#)文字や(.)の意味は次の通り。

- (1) # : 数値の各桁の位置を指定する。先行する0はスペースとなる。指定桁未滿は四捨五入して出力する。
- (2) . : 小数点の位置を指定する。
- 2 数値は、整数、単精度、倍精度のすべてが使用できる。
- 3 数値が指定桁数より長い場合、数値の前に%が出力される。
- 4 フォーマット文字列中には、前頁(その1)の文字列指定とまざってもよい。出力する変数の型の順序はそれに合わせる。

プログラム例: 行番号60は、フォーマット文字列で複数の書式を指定した例。70行は、フォーマット文字列中に、書式文字以外の文字を入れた例。

```

10 DEFINT N
20 DEFDBL D
30 N=123
40 A=4.56
50 D=3.14159265358979#
60 PRINT USING "####.###.#####"; N, A, D
70 PRINT USING "pai=###.#####"; D
80 END

```

実行例

```

123      4.6      3.1415927
pai= 3.1415926536

```

参照: PRINT, PRINT #n USING, LPRINT USING

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・数値が単精度変数の場合、
本文と同じ。高精度変数の場合、
指定桁数未滿を切り捨てる。
・数値が指定桁数より長い場合、(*)
が指定された桁だけ出力される。

MZK PRINT USING "###.##"; A :
PRINT USING "###.##"; A1
・1行には一つの指定のみ。
・DISK版のみ。

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・(1), (2)は本文と同じ。
・(#)の代わりに(Z)とすれば、
先行する0をそのまま出力できる。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・(1)は、(#)の代わりに
(Z)とすれば、同じことができる。
先行する0を出力するには、(Z)
を(9)とすればよい。指定桁未
滿は切り捨てる。

M243 ・M223と同じ。

MZB ・(1), (2)は本文と同じ。
ただし、指定桁未滿は切り捨てる。
・出力の型式指定は、一つしかでき
ない。複数必要なときは、PRINT
USING 文の後にセミコロン(:)
をつけながら、複数行 PRINT

USING 文を書く。
・上記の制限の替わりに、一連の数字
をとびとびに出力させることが
できる。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・数値は倍々精度も使用で
きる。SMC ・特徴3は ^ ^ が表示さ
れる。MZ35 ・特徴1(1)は切り捨てと
なり、特徴(3)はエラー。

X1Hu ・_ (アンダースコア) の
後に伴った書式指定子1個は文字
として表示する。
・特徴3は FORMAT OVER のエ
ラーが生ずる。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

書式指定プリント (その3) PRINT USING

形式: PRINT USING "+### +####"; A, A1

*省略・
繰り返し可

機能: 印字する数値への+の符号のつけかたを指定する。

特徴: 1 フォーマット文字列中の一連の、#の前後につけた+-の意味は次のとおり。

- (1) + : 数値の符号を付ける (正と0の場合は+, 負の場合は-)。▼形式(a) "#.##+" 数値の右端に符号を付ける。
(b) "+.##." 数値の左端に符号を付ける。
- (2) - : 負の数値に符号を付ける (正と0の場合はなし、負の場合のみ-)。▼形式(a) "#.##-" 負の数値の右端に-を付ける。(b)先頭に-を付けたときは、そのまま印字されるので、正の数値にも-が付いてしまう。
- 2 (1), (2)の指定と、まぜて使うことができる。
- 3 符号指定をしないと、正と0の場合はなし、負の場合は先頭に-がつく。

プログラム例: 種々の符号指定を行って印字させる。

```
10 A1=123:A2=-123
20 PRINT USING " #####"; A1; A2
30 PRINT USING " +#### "; A1; A2
40 PRINT USING " #####-"; A1; A2
50 PRINT USING " #### "; A1; A2
70 END
```

実行例

```
123+ 123-
+123 -123
123 123-
123 -123
```

参照: PRINT, PRINT #n USING, LPRINT USING

プリント・ユージング

APL —

TRS₁ : 本文と同じ。

CPM : 本文と同じ。

M223 : (1), (2)は、高精度数のみ使用できる。本文とは機能が異なる。

・(1)の(a)は本文と同じ。(b)は数値フィールド[(#)で示される部分]の左に符号が付く。また、"+.##"はPCにない機能である。このとき、(+)の数だけ、領域を確保する。また、(#)で示された桁より小さい数の場合、その部分はスペースでなく0となり、符号は(#)の直前の(+)の位置に付く。(b)で示された桁より大きく、(+)で示された桁より小さい数の場合は、先行する0はスペースとなり、数値の直前に符号が付く。

・(2)の(a)は本文と同じ。(b)“-##”とすれば、負の数の場合、数値フィールドの左に-が出力される。また、“-.-##”は、負の数の場合M223の(1)の“+.##”と同じ。ただし、正の場合符号は付かない。

MZK : DISK版のみ。MZBと同じ。

CBM —

PC8 : 本文参照。

TRS_{II} : 本文と同じ。

PC3 : (1)の(b)は本文と同じ。
・(2)の(b)“-##”とすれば、負の数の場合、数値の直前に符号が付く。

IF8₂ : 本文と同じ。レベル₃ : 本文と同じ。

C180 : (1)の(a)は“99+”で同様のことができる。ただし、先行する0も出力する。
・(2)の(b)“++9”で同様のことができる。+も領域を確保する。9で示された桁より小さい数は、その桁内の先行する0は0のままで、符号はその直前に付く。+で示される桁は、先行する0をスペースにし、数の直前に符号が付く。

M243 : M223と同じ。

MZB : (1)は形式を“+##”とする。
・(2)は形式を“-##”とする。

BUB : 本文と同じ。

FM8 : 本文と同じ。

PSP : (1)(b)はM223と同じ。
・(2)(b)はM223と同じ。

PC88 : 本文と同じ。

N52 : 本文と同じ。

PC6 —

MLT : 本文と同じ。

HC : 本文と同じ。

FP11 : 本文と同じ。

SMC : 先頭に-を付けたときも正の場合は符号がつかない。

MZ35 : 本文と同じ。ただし、特徴1(2)(b)で正の値には-はつかない。特徴3では符号を指定しないと符号はつかない。

書式指定プリント (その4) PRINT USING

プリント・ユージング

形式: PRINT USING " * * ### ¥ ¥ ###, "; A, A1

フォーマット文字列 *省略・
繰り返し可

機能: 数値の前のスペースを(*)にしたり, 数値の前に(¥)を付けたり, 3桁区切りで出力することなどを指定する。

特徴: 1 フォーマット文字列中の(**), (¥¥), (**¥), (,) の意味は次のとおり。

- (1) ** : 数値領域のはじめのスペースを(*)にする。**は2桁分の領域を確保する(ただし,(¥)に1桁使用する)。▼形式 " * * ###"
 - (2) ¥ ¥ : 出力する数値の直前のスペースの代わりに, 円記号(¥)を出力する。
¥ ¥ は2桁分の領域を確保する。
▼形式 " ¥ ¥ ###"
 - (3) ** ¥ : (1)と(2)の両方の機能をする。
3桁分の領域を確保する(ただし,(¥)に1桁使用する)。
▼形式 " ** ¥ ###"
 - (4) , : フォーマット文の小数点の左側にコンマ(,)がある場合, 整数部分の3桁ごとにコンマが出力される。
(,)は1桁分の領域を確保する。
- 2 金額を出力するとき便利である。

プログラム例: 行番号40~70は, (1)~(4)の例で, 80行は, それらを組み合わせて使用したものである。

```
10 A1=12:A2=12345
20 D#=1234567890#
30 PRINT USING "##### "A1:A2
40 PRINT USING "¥¥##### "A1:A2
50 PRINT USING "***¥### "A1:A2
60 PRINT USING "##### "A1:A2
70 E$="**¥,#####¥YEN "
80 PRINT USING E$:A1:D#
90 END
```

実行例

```
*****12    *12345
     ¥12    ¥12345
*****¥12    ¥12345
         12    12,345
*****¥12YEN ¥1,234,567,890YEN
```

参照: PRINT, RRINT #n USING, LPRINT USING

APL —

TRS_I ・(1)と(4)は本文と同じ。
・(2)は, 形式 " \$ \$ #"で同じことができる。ただし, 数値の直前には(¥)でなく(\$)を出力する。
・(3)は, 形式 " ** \$ #"で同じことができる。ただし, 数値の直前には(¥)でなく(\$)を出力する。

CPM ・TRS_Iと同じ。

M223 ・(2), (4)は高精度数のみ使用できる。
・(2)は, 形式 " ¥ ¥ #"で同じことができる。ただし, 数値が(#)で示された桁より小さい場合, (¥)は(¥)で示された桁の右端に付く。数値が(¥)で示された桁内の場合, 有効桁の直前に(¥)が付く。形式を " \$ \$ #"とすれば, (\$)を数値の前に出力することでもできる。
・(4)は本文と異なり, (,)で示された位置に(,)を出力するので, 3桁以外で区切ることもできる。
(,)は1桁分領域を確保する。

MZK ・DISK版のみ。MZBと同じ。

CBM —

PC8 ・本文参照。

TRS_{II} ・TRS_Iと同じ。

PC3 ・(1)は" **"で同じことができる。(*)は1桁分の領域を確保する。
・(2)は " ¥ #"で同じことができる。
・(4)は M223 の (4) と同じ。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・(1)はPC3の(1)と同じ。
・(2)は " ¥ ¥ ¥ Z "で同じことができる。また, " \$ \$ \$ Z "で(\$)を数値の直前に出力することができる。
・(4)は M223 の (4) と同じ。

M243 ・M223と同じ。

MZB ・(1)は本文の(1)と同じ。
・(2)はPC3の(2)と同じ。
・(4)は M223 の (4) と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・(¥)は指数形式で表示することはできない。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文のほかに形式 " \$ \$, ** \$ "もある。
・その機能は TRS_I の項参照。

SMC ・(1)は(*)で同じことができる。
・(2)は(¥¥)で同じことができる。
・(3)は(*¥)で同じことができる。
・(4)は本文と同じ。

MZ35 ・本文と同じ。特徴1(2)

書式指定プリント（その4） PRINT USING

プリント・ユージング

<p>は不可、Yは一つだけ使える。 (3)は不可、(3)では3桁ごとに ,を入れなければならない。</p>			
<p>X1Hu ・本文と同じ。</p>			
<p>MB16 ・本文と同じ。</p>			
<p>IBM55 ・本文と同じ。</p>			
<p>PC100 ・本文と同じ。</p>			
<p>MSX ・本文と同じ。</p>			

書式指定プリント (その5) PRINT USING

形式: PRINT USING "##. ###^/^/^/^"; A, A1

フォーマット文字列

*省略・
繰り返し可

機能: 指数型式で数値を出力する。

特徴: 1 フォーマット文字列中の一連の#の後につけた^/^/^/^は、次の意味をもつ。

^/^/^/^: 指数形式で数値を出力する。

▼形式 "##. #^/^/^/^"

□ □
* *

2 浮動小数点で表されるような、科学技術計算の結果の印字に便利である。

プログラム例: 行番号40は、指数型式での出力例。50~70は、符号のフォーマットと指数表示を組み合わせたもの。

```

10 DEFDBL D
20 A1=12345:A2=-3.14:A3=6.789E-20
30 D=1234567890123456#
40 PRINTUSING"##.###^/^/^/^";A1:A2:A3:D
50 PRINTUSING"+#.###^/^/^/^";A1:A2:A3:D
60 PRINTUSING"#.###^/^/^/^";A1:A2:A3:D
70 PRINTUSING"##.###^/^/^/^+";A1:A2:A3:D
80 END

```

実行例

```

1.23E+04 -3.14E+00 6.79E-20 1.23D+15
+1.23E+04 -3.14E+00 +6.79E-20 +1.23D+15
1.23E+04 3.14E+00- 6.79E-20 1.23D+15
1.23E+04+ 3.14E+00- 6.79E-20+ 1.23D+15+

```

参照: PRINT, PRINT #n USING, LPRINT USING

プリント・ユージング

APL —

TRS₁ ・ ^/^/^/^は↑↑↑↑で同じことができる。形式は、"##. #↑↑↑↑"となる。

CPM ・ 本文と同じ。

M223 ・ 本文と同じ。単精度数でも使用できる。

MZK —

CBM —

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 本文と形式が異なる。形式は、"##. #E #Z"で指数表示となる。本文の機能以外に、

- ・ %: %の次の文字を対応する変数の値だけ出力する。
- ・ X: スペースを出力する。nXとすれば、n個のスペースを出力する。
- ・ L: CR/LFを指定する。
- ・ C: CRを出力する。
- ・ N: ナル・コードを出力する。
- ・ T: TABコードを出力する。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ 形式 "Z9.9EEEE"で同じことができる。ただし、指数部の先行する0は0のままである。

- ・ 本文の機能以外に、フォーマット文字列の左端に(<)を記入すれば、左詰めで出力し、(>)を記入すれば、右詰めで出力する。

M243 ・ M223と同じ。

MZB —

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

- ・ 本文の機能以外に、(—)に続く1文字は、単なる文字として出力できる。制御文字を出力するときに使うと便利である。

N52 ・ 本文と同じ。

PC6 —

MLT ・ PC88と同じ。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。

SMC —

MZ35 ・ PC3と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ PC88と同じ。

IBM55 ・ PC88と同じ。

PC100 ・ PC88と同じ。

MSX ・ 本文と同じ。

プリンタへの出力 LPRINT

エル・プリント

形式：LPRINT A, “タテ” ; B

省略可 *省略・繰り返し可
リスト

機能：プリンタに情報を出力する。

特徴：1 省略形（L?）は使用できない。
2 その他は PRINT の項と同じ。

プログラム例：文字列をいろいろな形式でプリント出力する。

```
10 A$="タテ"
20 B$="ヨコ"
30 LPRINT A$
40 LPRINT B$
50 LPRINT
60 LPRINT A$;B$
70 END
```

実行例

タテ
ヨコ

タテヨコ

参照：PRINT

APL PR#6

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 ・CBM と同じ。

MZK PRINT/P, A, “タテ”

CBM PRINT#n, A, “タテ”

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 PRINT A, “タテ”

IF8₂ ・本文と同じ。レベル₃ ・PSP と同じ。

C180 ・OUTPUT#を用いる。

M243 ・CBM と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・PSP と同じ。

PSP PRINT#1: A, “タテ”

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC PRINT @1, A, “タテ” ; B

MZ35 PRINT A, “タテ” ; B

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

書式指定プリンタ出力 LPRINT USING

エル・プリント・ユージング

形式：LPRINT USING E1\$; A, E2\$

フォーマット¹⁾文字列 *省略・繰り返し可
(,は;でもよい)

機能：数値や文字列を、フォーマット文字列で指定した書式でプリンタに出力する。

特徴：1 プリンタに出力することを除いて、PRINT USING と同じである。
2 事務計算の結果のプリンタ出力に使うと便利である。

用語：¹⁾フォーマット 書式。入出力の際のデータの型式や配列、位置を示すもの。この場合はプリンタに出力する際の、位置や付け加える記号を指定するもの。
²⁾チャネル 一般に入出力装置と記憶装置との間で CPU の指令にもとづいて CPU と独立に直接情報の授受を行うための装置をいう。ここではポートと同義。

プログラム例：LPRINT USING を使った例。

```
100 DEFDBL D
110 E1$="PERSONAL":E2$="COMPUTER"
120 LPRINT USING "!:":E1$:E2$
130 A1=-12:A2=12345
140 D=1234567890#
150 LPRINT USING "#####.#+ ":A1:A2
160 LPRINT USING "#####.- ":A1:A2
170 LPRINT USING "+#####.# ":A1:A2
180 LPRINT USING "#####.# ":A1:A2
190 E$="***#.#####YEN "
200 LPRINT USING E$:A1:D
210 END
```

実行例

```
PC
      12.0-    12345.0+
****12.0-    *12345.0*
      -¥12.0  +¥12345.0
****-¥12.0  ¥12345.0
*****-¥12YEN ¥1,234,567,890YEN
```

参照：PRINT USING, PRINT #n USING

APL —

TRS₁ ・本文と異なる点はPRINT USING 参照。

CPM ・本文と異なる点はPRINT USING 参照。

M223 ・PRINT #6 USING で同様のことが可。チャネル²⁾（ここでは6）にプリンタ SOUT または SOUTA を指定する必要がある。本文と異なる点はPRINT USING 参照。

MZK PRINT/P USING "###"; A;
・1行に一つの指定のみ。
・DISK 版のみ。

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と異なる点はPRINT USING 参照。

PC3 ・PRINT USING で同様のことが可。本文と異なる点はPRINT USING 参照。

IF8₂ ・本文と同じ。

レベル₃ OPEN "O", #3, "LPT0:"
としておいてから、PRINT #3,
USING 文を使う。

C180 ・OUTPUT #6 USING で同様のことが可。本文と異なる点はPRINT USING 参照。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・レベル₃ と同じ。

PSP ・OPEN # "PRI", "X", 3,
"O" としておいてから、PRINT
#6 USING で同様のことが可。本文
と異なる点は PRINT USING
参照。

PC88 ・本文と異なる点はPRINT
USING 参照。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と異なる点はPRINT
USING 参照。

SMC PRINT @1, USING E\$,
A, E\$
・本文と異なる点は PRINT USING
参照。

MZ35 ・FP11 と同じ。

X1Hu ・PC88 と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・PC88 と同じ。

MSX ・本文と同じ。

画面のコピー COPY

形式: COPY M

↑
必要

機能: CRT に表示中の画面をプリンタにコピーする。

特徴: 1 M の値は 1 から 5 で、コピーする画面を示す。

- (a): テキスト画面のみ。
 (b): グラフィック画面のみ。
 (c): テキスト画面とグラフィック画面。
 (d): グラフィック画面漢字用 (上下が 1/2 の長さになる)。
 (e): テキスト画面とグラフィック画面 (上下が 1/2 の長さになる)。

2 プリンタによっては、コピーができない機種がある。

3 (COPY) キーでもプリントする。

用語: 1 テキスト(テキスト画面) キーボードから入力した文字や、プログラムのリストおよび実行結果を表示する画面。

実行例: 四角形を三つ描き、プログラム・リストとともにプリンタにハード・コピーをとる。

```

1 list
10 FOR I=1 TO 7
20 COLOR=(I,7-I)
30 NEXT I
40 LINE(50,50)-STEP(50,50),4,B
50 LINE(120,10)-STEP(90,80),5,B
60 LINE(10,5)-STEP(50,30),3,B
70 FOR I=1 TO 7
80 COLOR=(I,1)
90 NEXT I
100 COPY 5
110 END
Ok
run

```

参照:

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 — 124A 番地を call すると
表示画面をコピーする。TRS_{II} —

PC3 —

IF8₂ COPY
 ・ディスプレイに表示されている画像のうち、バック・グラウンド・カラーと違う色の部分をプリンタに印刷する。

レベル₃ —

C180 (REQ) CP (NL) ができる。

M243 —

MZB COPY/P 1

BUB —

FM8 HARDC M1
 ・M1=0 でキャラクタ, M1=1 でグラフィックス (大), M1=2 でグラフィックス (小)。

PSP —

PC88 — 本文参照。

N52 —

PC6 LCOPY

・プリンタは画面と同じ絵柄を出力する。

MLT — PC6 と同じ。

HC COPY

・LCD に表示されている文字・図形を内蔵プリンタに出力する。

FP11 COPY M1, M2

・画面の横方向がプリンタの上下方向に対応する。

SMC —

MZ35 GPRINT N1, N2, N3, N4, N5

・(N1, N2) (左上), (N3, N4) (右下) を対角線とする四角形の CRT 上に表示されたグラフィック・パターンを出力開始位置 N5 よりプリンタに出力する。

X1Hu HCOPY M

・M は 0 ~ 4 の整数。
 0: グラフィック 1, 2, 3 の画面の合成
 1: グラフィック 1 の画面
 2: グラフィック 2 の画面
 3: グラフィック 3 の画面
 4: テキストとグラフィック 1, 2, 3 の画面の合成
 省略時はテキスト画面。

MB16 — コマンドではない、ただし、特殊キーの機能としてある。

IBM55 — コピー・キーがある。

PC100 — コピー・キーのほかに LCOPY でプリンタに出力できる。

MSX —

シーケンシャル・ファイル出力文 PRINT #n

プリント #n

形式: PRINT #6, A, C#; E\$;

*省略・繰り返し可

機能: シーケンシャル・ファイルにデータを出力する。

- 特徴: 1 ファイル番号は, OPEN 命令ですでにオープンされたものでなければならぬ。
 2 (,), (;) の意味は PRINT 命令と同じ。
 3 データをプリンタに打ち出すようなイメージでディスクケットに書きこむとき使われる。

プログラム例: ファイル"AFILE"を作り, 書き込み, 読み出す。

```
10 PRINT "make & open AFILE":OPEN
  "AFILE" FOR OUTPUT AS #5
20 PRINT #5,"TEST PROGRAM"
30 PRINT "close AFILE":CLOSE 5
40 PRINT "open AFILE":OPEN "AFILE"
  FOR INPUT AS #6
50 INPUT #6,E$
60 PRINT E$
70 PRINT "close AFILE":CLOSE 6
80 END
```

実行例

```
make & open AFILE
close AFILE
open AFILE
TEST PROGRAM
close AFILE
```

参照: OPEN, PRINT, WRITE#

```
APL D$ = CHR$(4)
PRINT D$;"WRITE AEILE"
PRINT A, E$
```

TRS_I ・本文と同じ。

CPM ・本文と同じ。ただし, カセット・レコーダは扱えない。

M223 ・CPM と同じ。

MZK ・CPM と同じ。

CBM ・本文と同じ。ただし, カセット・レコーダへの出力は, OPEN 文でカセット・ファイルをオープンして用いる。

PC8 ・本文参照。
・ファイル番号を-1にするとカセット・レコーダに出力する。TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。ただし, カセット・レコーダへの出力は, CSAVE 命令を用いる。

IF82 ・CBM と同じ。

レベル₃ ・CBM と同じ。C180 ・可変長の場合 OUTPUT #1:A,
・固定長の場合 PUT #1:A,

M243 ・CBM と同じ。

MZB ・本文と同じ。ただし, カセット・レコーダへの出力は, PRINT/T,

BUB ・本文と同じ。

FM8 ・CBM と同じ。

PSP PUT #1, A, E\$
・機能は CPM と同じ。

PC88 ・本文と同じ。

N52 ・(")を書き込む時は CHR\$(34)を使用する。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・CAS0, CAS1(カセット)に対しては " " と " ; " のどちらでも同じ形式で出力される。

FP11 ・本文と同じ。

SMC ・ファイルは OPEN/O でオープンされたものに限る。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・(,) と (;) の区別がなく, 数式は (;) で区切る必要がある。

PC100 ・本文と同じ。

MSX ・本文と同じ。

書式指定ファイル出力 PRINT #n USING

プリント #n ユージング

形式: PRINT #5, USING E\$; A, A1

ファイル番号 フォーマット *省略・
文字列 繰り返し可

機能: 数値や文字列をフォーマット文字列で指定した書式で、ファイルに出力する。

特徴: 1 ファイルに出力することを除いて、PRINT USING と同じである。
2 文字列変数中の (,) は、データ区切り記号となるので、この場合 INPUT # で入力すると、(,) の前後でデータが分割されるので注意、(,) が必要なきは、LINE INPUT # を使うとよい。

プログラム例: PRINT #n, USING でファイル出力し、そのファイルを LINE INPUT # で入力し、プリントする。

```
100 OPEN "AFILE" FOR OUTPUT AS #5
110 E1$="PERSONAL":E2$="COMPUTER"
120 PRINT#5,USING "!:":E1$;E2$
130 PRINT#5,USING "& &-& &":E1$;E2$
140 A1=123456!:A2=-4.56
150 PRINT #5,USING "#####.# ":A1;A2
160 PRINT #5,USING "***#####.# ":A1;A2
170 E$=" ***#,#####YEN "
180 PRINT #5,USING E$;A1;A2
190 CLOSE
200 OPEN "AFILE" FOR INPUT AS #6
210 IF EOF(6) THEN 240 ELSE LINE INPUT #6,E$
220 LPRINT E$
230 GOTO 210
240 CLOSE
250 END
```

実行例

```
PC
PER-COM
123456.0                  -4.6
¥123456.0      *****¥4.6
***¥123,456YEN      *****¥5YEN
```

参照: PRINT USING, LPRINT USING

APL —

TRS₁ ・ 本文と異なる点は、
PRINT USING 参照。CPM ・ 本文と異なる点は、
PRINT USING 参照。M223 ・ 本文と異なる点は、
PRINT USING 参照。

MZK —

CBM —

PC8 ・ 本文参照。

TRS_{II} ・ 本文と異なる点は、
PRINT USING 参照。PC3 ・ PRINT #5 USING で同
様のことができる。本文と異なる
点は、PRINT USING 参照。IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。C180 ・ OUTPUT #5 USING で
同様のことができる。本文と異な
る点は、PRINT USING 参照。

M243 ・ M223 と同じ。

MZB —

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ WRITE #5 USING で
同様のことができる。本文と異な
る点は、PRINT USING 参照。

PC88 ・ 本文と異なる点は、

PRINT USING 参照。

N52 ・ 本文と同じ。

PC6 —

MLT ・ PC88 と同じ。

HC ・ 特徴 2 は不可。
・ データとデータの間にはユーザが
指定しない限り区切り記号は書か
れない。FP11 ・ 本文と異なる点はPRINT
USING参照。SMC ・ FP11 と同じ。なお、フ
ァイルは OPEN/O でオープンさ
れたもの。

MZ35 —

X1Hu —

MB16 ・ 本文と同じ。

IBM55 ・ 特徴 2 は (,) と (;)。

PC100 ・ PC88 と同じ。

MSX ・ 本文と同じ。

ランダム・ファイルへの出力 PUT

プット

形式: PUT #9, 3

□ □

省略可

機能: ランダム・ファイルの指定された1レコードへ書き込む。

- 特徴: 1 この命令を実行する前に、OPEN 文、FIELD 文を実行しなければならない。
- 2 書き込むデータは、FIELD 文で指定した変数にあらかじめ代入しておく。
- 3 3はレコード番号。これを省略すると最後のGET 文またはPUT 文の次のレコードが指定される。
- 4 ランダム・ファイルは、データ・ファイルの途中から引き出す必要のあるデータの格納に便利であり、その際にはこの命令を使う。

プログラム例: ランダム・ファイル“AFILE”の作成、書き込み、読み出し。

```
10 OPEN "AFILE" AS #9
20 FIELD #9,20 AS E$
30 FOR I=1 TO 5
40 LSET E$="test data"+STR$(I)
50 PUT #9,I
60 NEXT I
70 FOR I=5 TO 1 STEP -1
80 GET #9,I
90 PRINT E$;
100 NEXT I
110 CLOSE 9
120 END
```

実行例

```
test data 5    test data 4    test data 3
test data 2    test data 1
```

```
APL D$=CHR$(4): PRINT
D$;"WRITE AFILE,R";3:
PRINT A,E$
```

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 PUT #1 RECORD 3

```
MZK PRINT #1 (3), A, E$
・1レコードは32バイトの固定長。
```

```
CBM ・RECORD 命令でポイン
タを設定した後、PRINT #1, A,
E$。
```

PC8 ・本文参照。

```
TRS11 PUT 1, 25
・1はレコード番号。25はレコード
番号。
```

PC3 ・本文と同じ。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 WRITE #9: 3, E\$

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・PUT%とするとRAM フ
ァイルに変数値を書き込める。例
えば、PUT%0, A/とすると、
レコード番号0へA/の値を書き
込むことを意味する。

FP11 ・本文と同じ。

SMC ・FIELD 文に相当するも
のは、RECORD 文である。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。

MB16 ・レコード番号は、1から
(ディスクの未使用バイト数/レコ
ード長)まで。ただし、32767より
小さい。

IBM55 ・レコード番号は1から
32767まで。

PC100 ・本文と同じ。

MSX —

参照: OPEN, GET, FIELD, PRINT #n

フィールド内左づめ書き込み LSET

レフト・セット

形式: LSET E1\$ = "ABCDE"

機能: ランダム・ファイル・バッファ¹⁾のフィールド²⁾に文字列を左づめで書き込む。

- 特徴: 1 フィールドが文字列長より長いとき、左づめでランダム・ファイル・バッファに書き込み、空き領域に空白をつめる。
 2 文字列がフィールド長より長いとき、右側にはみ出た文字は無視される。
 3 このコマンドの前に、FIELD 文を実行しなくてはならない。
 4 数値はこのコマンドを実行する前に、MKI\$, MKS\$, MKD\$などで文字に変換しなければならない。
 5 ランダム・ファイルに書き込むとき用いる。

用語: ¹⁾バッファ データ処理のために、データを一時蓄えておく領域。
²⁾フィールド 1レコード分のファイル情報が定義される場所のこと。

プログラム例: フィールド文でフィールド構成を決めて、名前を左づめで書き込みそのまま出力する。

```
100 OPEN "KYU-f" AS #1
110 FIELD #1,15 AS N$,5 AS S$
120 INPUT "NAME";I$
130 IF I$="E" THEN 190
140 LSET N$=I$
150 INPUT "YOKINZANDAKA";A$
160 RSET S$=A$
170 PUT #1
180 GOTO 120
190 LPRINT " ** YOKIN ZANDAKA **"
200 FOR I=1 TO LOF(1)
210 GET #1,I
220 LPRINT N$;S$;"R"
230 NEXT
240 NAME "KYU-f" AS "SHIN-f"
250 CLOSE:END
```

実行例

```
** YOKIN ZANDAKA **
miyazaki      5000円
A.nakamura    8900円
nakanishi     10000円
noda          10000円
nishida       7500円
horiuchi      15000円
suzuki        6000円
Y.nakamura    10000円
```

参照: RSET, FIELD, PUT, GET

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。また、ある領域内に、ストリングを左づめにするために使うこともできる（このときは、前もって FIELD 文を実行しなくてもよい）。

M223 ・該当なし。配列文字変数 RECORDSIZ 文で設定すると、自動的にチャンネル・バッファに左づめで書き込まれる。

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・該当なし。ただし、E\$ = "ABC" でレコード変数に文字変数を入力すると左づめにはできる。

M243 ・M223と同じ

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。ただし、カ

セット・テープに対しても同じことができる。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・CPMと同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC LJUST 20 AS ES
 ・RECORD 文中で指定する。
 ・数値は文字長で変数、式も使用できる。

MZ35 —

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。ただし、FIELD 文で定義されなかった文字変数に対しても、与えられたフィールド内で左寄せができる。

PC100 ・本文と同じ。

MSX —

フィールド内右づめ書き込み RSET

ライト・セット

形式: RSET E1\$ = "ABCDE"

機能: ランダム・ファイル・バッファのフィールドに文字列を右づめで書き込む。

特徴: 1 フィールド長が文字列長より長いとき、右づめでランダム・ファイル・バッファに書き込まれる。ほかは LSET と同様。
 2 数値など、右にそろえて出力したい時使うと便利である。
 3 ランダム・ファイルに書き込む時利用。

プログラム例: フィールド文でフィールド構成を決めて、預金残高を右づめて書き込み、そのまま出力する。

```

100 OPEN "KYU-f" AS #1
110 FIELD #1,15 AS N$,5 AS S$
120 INPUT "NAME"; I$
130 IF I$="E" THEN 190
140 LSET N$=I$
150 INPUT "YOKINZANDAKA"; A$
160 RSET S$=A$
170 PUT #1
180 GOTO 120
190 LPRINT " ** YOKIN ZANDAKA **"
200 FOR I=1 TO LOF(1)
210 GET #1, I
220 LPRINT N$; S$; "R"
230 NEXT
240 NAME "KYU-f" AS "SHIN-f"
250 CLOSE:END

```

実行例

```

** YOKIN ZANDAKA **
akai          10000R
okuda         1200R
nakajima      5000R
fuwa          15000R
takeuchi      5030R
hino          7500R
yoshizawa     32000R
Y.nakamura    10000R

```

参照: LSET, FIELD, PUT, GET

APL —

TRS_I ・本文と同じ。

CPM ・本文と同じ。また、ある領域内で、ストリングを右づめにするために使うこともできる（このときは、前もって FIELD 文を実行しなくてもよい）。

M223 ・該当なし。ただし、配列数値変数は、RECORDSIZ 文で設定すると、自動的にチャンネル・バッファに右づめて書き込まれる。

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・該当なし。ただし、MOVE E\$ = 123.45 [2] でレコード変数に数値を代入すると、自動的に右づめにはできる。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。なお、カセ

ットに対しても同じことができる。

NZ52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・CPM と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC RJUST 20 AS FS (2)
 ・RECORD 文中で指定する。
 ・数値は文字長で変数、式も使用できる。

MZ35 —

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。ただし、FIELD 文で定義されなかった文字変数に対しても、与えられたフィールド内で右寄せができる。

PC100 ・本文と同じ。

MSX —

内部コードのファイルへの出力 WRITE #n

ライト #n

形式: WRITE #6, A, C#, E\$;

*省略・繰り返し可

機能: シーケンシャル・ファイルに、変数や定数の内部コード¹⁾を出力する。

- 特徴: 1 ファイル番号は、OPEN 命令ですでにオープンされたものでなければならない。
 2 内部コードは、文字列データについては文字数+1バイト、単精度数値は4バイト、高精度数値は9バイトとなる。
 3 文字変数は、指定文字数分一杯に文字を入れておかねばならない。
 4 PRINT #n に比べて、使用領域が少なくて済む。
 5 (;) はフォーマットをそろえるために必要である。

用語: ¹⁾内部コード 各値はメモリ中で2進型式で表されているが、その型式のデータをいう。

プログラム例: 内部コードでファイルに書き込み、読み出す。

```

10 OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20 OPEN "AFILE" FOR OUTPUT AS FILE 6
30 FOR I = 1 TO 5
40   WRITE #6, I
50 NEXT I
60 CLOSE 6
70 OPEN "AFILE" FOR INPUT AS FILE 5
80 FOR I = 1 TO 5
90   READ #5, A
100  PRINT #1, "DATA=" ; A
110 NEXT I
120 CLOSE 5
130 CLOSE 1
140 END

```

実行例

```

DATA= 1
DATA= 2
DATA= 3
DATA= 4
DATA= 5

```

参照: OPEN, READ #n, PRINT #n

APL —

TRS_I —

CPM —

M223 ・本文参照。

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 BPRINT #1, A, E\$

IF8₂ —レベル₃ —

C180 —

M243 ・本文と同じ。

MZB —

BUB —

FM8 —

PSP —

PC88 ・本文と同じ。

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC SEND #6, A, E\$

- ・ファイルは OPEN/S で既にオープンされたものでなければいけない。
- ・ファイル番号は 0 から 127。

MZ35 ・PC3 と同じ。

X1Hu ・データの区切りに(,)を入れ、文字列データのときは(")をつけ、詰めて出力する。

MB16 ・本文と同じ。

IBM55 ・特徴 1、4 以外はない。
 ・変数は(,) (;) で区切る。

PC100 ・(;) は必要なし。

MSX —

ポート出力 OUT

アウト

形式：OUT32, &H64

機能：指定した出力ポートに、1バイトのデータを出力する。

- 特徴：1 初めの引数は、出力ポートの番号、第2の引数は、出力するデータを示す。
 2 引数の範囲は、0～255までの10進整数または &H00～&HFF までの16進数である。
 3 外部 I/O 装置とのインターフェースに使う。

プログラム例：A～Z までのアルファベットのアスキーコードを OUT 命令でプリンタに送り、13文字ごとに改行コードを送ってプリンタから出力する。

```
10 FOR J=65 TO 81 STEP 13
20 FOR I=J TO J+12
30 GOSUB 100
40 NEXT I
50 I=10:GOSUB100
60 NEXT J
70 END
100 IF (INP(&H40) AND 1)=1 THEN GOTO 100
110 OUT &H10,I
120 OUT &H40,1
130 OUT &H40,0
140 RETURN
```

実行例

ABCDEFGHIJKLM
 NOPQRSTUVWXYZ

参照：INP

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。ただし、第1の引数で指定された装置番号に、第2の引数で指定された1バイト・データを出力する。

MZK ・本文と同じ。なお、出力ポート番号を10進にするときは前に@をつける。

CBM —

PC8 ・本文参照。

TRS₁₁ —

PC3 —

IF8₂ ・本文と同じ。ただし、出力ポート番号は、2バイト（-32768～32768）である。

レベル₃ —

C180 —

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB —

FM8 —

PSP ・本文と同じ。16進数は\$を前に付ける。

PC88 ・本文と同じ。

N52 —

PC6 ・本文と同じ。

MLT ・初めの引数は0以上65535以下の整数式。
 ・&Hのほか、&O（8進）、&B（2進）も可。

HC —

FP11 ・出力ポート番号は、-32768から65535までの範囲で指定する。
 ・LOUT 文でデータをプリンタへ出力する。

SMC ・本文と同じ。

MZ35 —

X1Hu ・本文と同じ。ただし、出力ポート番号は0～&HFFFF。

MB16 ・ポート番号は0～65535。

IBM55 ・MB16と同じ。なお、引数は、0から255までの数式。

PC100 ・本文と同じ。

MSX ・本文と同じ。

プログラムのロード LOAD

形式：LOAD "1:AFILE", R

省略可

機能：ディスクから、指定したファイルの内容をメモリにロードし実行する。

- 特徴：1 LOADの指定は、ドライブ・ナンバ、()、ファイル名、()、R オプション。
 2 ファイル名は、SAVE コマンドで既に格納されたものでなければならぬ。
 3 R オプションを指定しないときは、ファイルの内容をロードする前に開いているファイルをすべて閉じ、メモリにあるすべての変数とプログラムを抹消する。
 4 R オプションを指定したときは、それまで開いていたファイルは開いたままで、指定したプログラムはロードした後自動的に実行される。
 5 プログラムに比べて使えるメモリのサイズが小さいとき、R オプションを指定して、いくつかのプログラムを連結して使用できる。このとき、プログラム間の情報は、これらのデータ・ファイルを通じて渡される。

実行例：フロッピー・ディスクに「EXAMPLE」の名で格納されていたプログラムをロードする。

```
list
10 INPUT "A=";A
20 IF A = 0 THEN GOTO 50
30 PRINT A;"^2=";A^2
40 GOTO 10
50 END
Ok
save "2:EXAMPLE"
Ok
new
Ok
load "2:EXAMPLE"
Ok
list
10 INPUT "A=";A
20 IF A = 0 THEN GOTO 50
30 PRINT A;"^2=";A^2
40 GOTO 10
50 END
Ok
```

参照：SAVE, RUN

APL LOAD "AFILE" S7, D1, V1

- ・指定順は、ファイル名、スロット・ナンバ、ドライブ・ナンバ、ボリューム・ナンバ、R オプションはない。

TRS_i ・本文と同じ。

CPM ・本文と同じ。

M223 OLD "1:AFILE"

- ・SAVE コマンドで格納したファイルのロードは、この OLD コマンドを用いる。

LOAD "1:AFILE"

- ・LIST コマンドで格納したファイルは、LOAD コマンドを用いる。

MZK LOAD FD1, "AFILE"

- ・機能は本文と同じ。ただし、R オプションはない。

CBM DLOAD "AFILE", D0 ON U8

- ・指定順は、ファイル名、ドライブ・ナンバ、ユニット・ナンバ、R オプションはない。

PC8 ・本文参照。

TRS_{ii} ・本文と同じ。

PC3 LOAD "AFILE", 100

- ・AFILE をロードした後、100 行目から実行する。
- ・起動時のプログラム自動読み込み可能な ALOAD もある。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。ただし、プログラムを見つける前に **(CTRL)** **-D** キーを入力すると実行を停止し、メモリ中のプログラムは保存

される。

C180 LOAD AFILE .SR/1

- ・ソース・ワークから入力中エラーが発生したとき、一時格納を中断した後、続けて格納したいとき LOAD のみを入力すればよい。
- ・NEW を実行したあとでロードしないと、プログラムがマージされる。

M243 ・M223 と同じ。

MZB LOAD FD1@1, "AFILE"

- ・指定順は、ドライブ・ナンバ、スレーブ・ディスク・ボリューム・ナンバ、ファイル名、R オプションはない。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP LOAD DF1, "AFILE", HIMITU, @

- ・指定順は装置番号、ファイル名、パスワード、(@)。
- ・@ を指定すると現在の LOMEM が使用される。

PC88 ・本文と同じ。ただし、ディスク以外にも使用できる。

N52 ・R オプションの指定がないときはコマンド・レベルにもどる。

PC6 ・本文と同じ。

MLT ・ドライブ・ナンバは A, B, C, D。

HC ・LOAD "COM0 : (BLPSC)" とすると、プログラムを RS-232C ポートより読み込む。

プログラムのロード LOAD

ロード

<p>・BLPSC は接続条件の設定。</p>			
<p>FP11 ・プリンタ以外のデバイスでも実行できる。</p>			
<p>SMC ・PC88 と同じ。ただし、R オプションは SAVE 時に指定できる。 ・LOAD/V “AFILE” でベリファイできる。 ・機械語プログラムが同時にセーブされていたなら同時にロードされる。 ・CLOAD も使用可能。</p>			
<p>MZ35 ・PC3 と同じ。 LOAD SUB “ASUB” ・ディスク上のプログラムをメモリ上のプログラムの後に接続してサブプログラムのに使う。</p>			
<p>X1Hu ・R オプションはない。</p>			
<p>MB16 ・PC88 と同じ。ただし、エクステンションが省略され、ファイル名が 8 文字以下では “.BAS” がつけられる。</p>			
<p>IBM55 ・MB16 と同じ。なお、LOAD “AFILE”, R は、RUN “AFILE” と同じ。</p>			
<p>PC100 ・本文と同じ。なお、拡張子 “.BAS” は省略できる。</p>			
<p>MSX ・現在ディスクはサポートされていないが、ドライブ・ナンバを “CAS” で置き換えることにより、カセット・テープからロードし、実行することができる。</p>			

プログラムの格納 SAVE

セーブ

形式: SAVE "1:AFILE", A

省略可

機能: 指定したファイル名で、メモリの内容をフロッピー・ディスクに格納する。

- 特徴: 1 SAVE の指定は、ドライブ・ナンバ、ファイル名、A オプションの順である。
- 2 指定したファイル名が、既に存在しているときは、古い内容を消して新しい内容を格納する。
- 3 A オプションを指定したときは、アスキー・コードで、指定しないときは、バイナリ・コード¹⁾でファイルを格納する。アスキー・コードで格納すると、バイナリより大きいディスク・スペースを必要とするが、コマンドによっては、アスキー・コードのファイルが必要とする (MERGE など)。
- 4 ドライブ・ナンバを省略したときは、最小の番号 (PC8 の場合は 1) を指定したことになる。

用語: ¹⁾バイナリ・コード メモリに記憶されている型式の情報。

実行例: メモリのプログラムを "EXAMPLE" の名でフロッピー・ディスクに格納し、次にそれをメモリにロードする。

```
list
10 INPUT "A=";A
20 IF A = 0 THEN GOTO 50
30 PRINT A;"^2 =" ;A^2
40 GOTO 10
50 END
Ok
save "2:EXAMPLE"
Ok
new
Ok
load "2:EXAMPLE"
Ok
list
10 INPUT "A=";A
20 IF A = 0 THEN GOTO 50
30 PRINT A;"^2 =" ;A^2
40 GOTO 10
50 END
Ok
```

参照: LOAD, MERGE, SET, STORE, KEYList

APL SAVE "AFILE", S7,
D1, V1

- ・指定順は、ファイル名、スロット・ナンバ、ドライブ・ナンバ、ボリューム・ナンバ。

TRS₁ SAVE "AFILE"/BAS
HIMITU: 1, A

- ・指定順は、ファイル名、(/)、拡張ファイル名、パスワード、(:), ドライブ・ナンバ、A オプション。

CPM SAVE "AFILE", A, P
・機能は本文と同じ。なお、P オプションをプログラム保護のため使用できる。M223 SAVE "1:AFILE"
・ドライブ1に中間言語で格納する。これをロードするときは、OLD による。
LIST "1:AFILE"
・ドライブ1にソース・コードで格納する。これをロードするときは LOAD による。

MZK ・本文参照。

CBM DSAVE "AFILE", D0
ON V8
・指定は、ファイル名、ドライブ・ナンバ、ユニット・ナンバ。

PC8 ・本文参照。

TRS₁₁ ・TRS₁ と同じ。PC3 SAVE @ "AFILE", 70,
170
・70行から170行までを格納する。170を省略すると、70行以降を、二つとも省略するとすべての行を格納する。プログラム保護のため (@) の指定ができる。

- ・アスキー型式で格納のときは、STORE を用いる。
- ・プログラム保護のため LOCK, UNLOCK 命令もある。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。C180 SAVE AFILE, SR/1
・ソース・ワークのとき SR を指定、コード・ワークのとき IB を指定、省略すると両方退避する。

M243 ・M223 と同じ。

MZB SAVE FD1@1, "AFILE"
・指定順は、ドライブ・ナンバ、スレーブ、ディスクセット・ボリューム・ナンバ、ファイル名。

BUB ・本文と同じ。

FM8 ・本文と同じ。最小のドライブ・ナンバは 0。

PSP SAVE DF1, "AFILE",
HIMITU, *
・指定順は、装置番号、ファイル名、パスワード、(*)。
・(*)をプログラム保護のため指定できる。

PC88 ・本文と同じ。

N52 ・A オプションを指定したときは、JIS コードで格納する。
・ドライブ・ナンバを省略したときは、0 を指定したことになる。

PC6 ・本文と同じ。

MLT SAVE "A:AFILE",
A | P
・P は保護ファイルとして格納する

プログラムの格納 SAVE

セーブ

<p>(省略可). LOAD, RUN はできるが, LIST, EDIT はできない. ドライブ・ナンバは省略可.</p> <hr/> <p>HC SAVE "B:TEST", A ・ A をつけないと, バイナリ型式でセーブされる. V オプションだと, マイクロ・カセットにセーブできる (SAVE "CAS0:ABC", V). ・ SAVE "COM0:(BLPSC)" とすると RS-232C ポートの接続条件 (BLPSC) を設定し, プログラムを出力する. ・ 次のようにも使える. LIST "B:TEST, ASC" アスキー型式の SAVE と同じもの. 格納するプログラムの行も指定できる.</p> <hr/> <p>FP11 ・ ドライブ名を省略すると 0 が指定される.</p> <hr/> <p>SMC SAVE "B:AFILE", 先頭アドレス, 最後アドレス. ・ 引用符の前に /A を付けると, アスキー型式で格納され, /P を付けるとリスト出力ができない形でセーブされる. /C を付けると任意のコマンドと同時にセーブできる. ・ ディスク以外にも使用できる. ・ 先頭アドレスと最後アドレスで指定されたアドレス間のメモリ内容が, BASIC プログラムと同時に格納される.</p> <hr/> <p>MZ35 ・ 本文と同じ. STORE ・ プログラムをアスキー型式でディスクへ書き込む.</p> <hr/> <p>X1Hu ・ FP11 と同じ.</p> <hr/> <p>MB16 ・ P オプションによりファイルは保護される. エクステンションが省略されファイル名が 8 文</p>	<p>字以下では ".BAS" が付く.</p> <hr/> <p>IBM55 ・ MB16 と同じ. なお, A オプションは文字型式でセーブされる.</p> <hr/> <p>PC100 ・ MB16 と同じ.</p> <hr/> <p>MSX SAVE "CAS:AFILE" ・ 常にアスキー・セーブされる.</p> <hr/>		
--	---	--	--

ファイル名変更 NAME

形式: NAME "AFILE" AS "BFILE"

旧ファイル名 新ファイル名

機能: ディスク・ファイル上の名前を変更する。

- 特徴: 1 旧ファイル名を次の新ファイル名に変える。
 2 旧ファイル名は、ディスク上には存在しなければならない。また、新ファイル名は、同一ディスク上には存在してはならない。
 3 このコマンドを実行すると、ファイル名だけが変わり、そのファイルのディスク上の位置は変わらない。

用語: ¹⁾ エクステンション 拡張子のこと。AFILE, BAS の .BAS がそれにあたり、ファイル名の一部である。

プログラム例: KYU-f に対して入出力を行った後、ファイル名を SHIN-f に変える。

```
100 OPEN "KYU-f" AS #1
110 FIELD #1,15 AS N$,5 AS S$
120 INPUT "NAME";I$
130 IF I$="E" THEN 190
140 LSET N$=I$
150 INPUT "YOKINZANDAKA";A$
160 RSET S$=A$
170 PUT #1
180 GOTO 120
190 LPRINT ** YOKIN ZANDAKA **
200 FOR I=1 TO LOF(1)
210 GET #1,I
220 LPRINT N$;S$;"P"
230 NEXT
240 NAME "KYU-f" AS "SHIN-f"
250 CLOSE:END
```

実行例

```
** YOKIN ZANDAKA **
akai          10000円
okuda         12000円
nakajima      50000円
fuwa          15000円
takeuchi     50300円
hino          75000円
yoshizawa    320000円
Y.nakamura   10000円
```

APL RENAME "AFILE", "BFILE"

- ・本文と同じ。ただし、このコマンドは新ファイル名が同じディスク上にあるかを検出しない。また、ディスク上の複数の同一ファイル名についてこのコマンドが使えない。

TRS1 RENAME "AFILE" TO "BFILE"

- ・本文と同じ。ただし、このコマンドはDOSコマンドで、DOSモードでのみ実行できる。

CPM ・本文と同じ。

M223 RENAME 1: AFILE_1: BFILE

- ・機能はTRS1と同じ。
- ・1:を省略するとドライブ0。
- ・オープン中のファイルには使用できない。

MZK RENAME FD1@1, "AF", "BF"

- ・ドライブ名、スレーブ・ディスク・ドライブ番号を指定できる(省略可)。
- ・ロックされているファイルには実行できない。ほかは本文と同じ。

CBM RENAME 1, "AF" TO "BF" ON 7

- ・オープン中のファイルには使用できない。
- ・最初の引数でドライブ番号を指定し、最後の引数でユニット番号を指定する。省略するとそれぞれ、0, 8に設定される。
- ・ファイル名に変数、番号も使える。そのときはカッコでくくる。

PC8 ・本文参照。

TRS1 RENAME "AFILE" AS "BFILE"

- ・本文と同じ。

PC3 RNAME "AF, SHARP: A1", "BF"

- ・旧ファイルにボリューム名、チャンネル・ドライブ番号を指定する。新ファイルでは、もし指定しても無視される。ほかは本文と同じ。

IF82 NAME "2: AF" AS "2: BF"

- ・旧ファイルのあるドライブが1のときは省略できるが、ほかのときは指定しなければならない。

レベル3 ・本文と同じ。

C180 RENAME A.B, D.B

- ・TRS1と同じ。ただし、種別の変更もできる。

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP RENAME "FD1", "AF", "BF"

- ・FD1(ドライブ番号)、旧ファイル名(AF)とそのパスワード、新ファイル名(BF)とそのパスワードの順で指定する。パスワードは省略できる。また引用符も省略できる。ほかは本文と同じ。

PC88 ・本文と同じ。

N52 ・ドライブ番号を省略すると0である。省略しない場合は、両方とも同じ値でなければならない。

参照: KILL, SET

ファイル名変更 NAME

ネーム

い.			
PC6 ・本文と同じ。ドライブ番号をつけられる。			
MLT ・IF8 ₂ と同じ。ドライブNo.はA, B, C, D.			
HC ・本文と同じ。			
FP11 ・IF8 ₂ と同じ。ただし、ドライブ名を省略すると、ドライブ0が指定される。			
・ファイル名の前に PACK0 : ~ PACK7 : を付けることにより、RAM バックにおいても実行可能。			
SMC REN "AFILE" TO "BFILE" ・機能は本文と同じ。			
MZ35 RNAME AFILE, BFILE ・機能は本文と同じ。			
X1Hu ・ファイル・ディスクリプタを省略すると、DEVICE 命令で指定したものになる。			
MB16 ・本文と同じ。			
IBM55 ・ドライブ・ナンバの指定ができる。 ・エクステンション ¹⁾ は省略できない。			
PC100 ・PC100は階層ディレクトリ構造をもつが、NAMEはカレント・ディレクトリに対してのみ有効。ファイル名の変更は、ファイルを閉じた状態で行うこと。			
MSX —			

ファイルの消去 KILL

形式：KILL “AFILE”

機能：指定したファイルを消去する。

- 特徴：1 書き込み禁止属性のつけられたファイルは消去できない。
 (書き込み禁止は、SET 命令で行える)
 2 消去するファイルはオープンされてはならない。
 3 いらぬファイルを消去するのに便利である。

実行例：“AFILE”を削除する。

```
files
test1. 1 test2. 1 AFILE. 1
Ok
kill"AFILE"
Ok
files
test1. 1 test2. 1
Ok
```

参照：SET, SAVE

APL DELETE AFILE

・機能は本文と同じ。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 DELETE AFILE

- ・消去できるファイルは拡張子が APU のもののみ。ほかのものを消去する場合 CLI モードで DELETE コマンドを実行。
- ・KILL 命令は変数を消去する時に用いられる。
- ・コンマで区切って消去するファイル名をならべることができる。

MZK DELETE “AFILE”

・機能は本文と同じ。

CBM SCRATCH “AFILE”

・機能は本文と同じ。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 ・本文と同じ。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・CLI モードで DELETE AFILE。プログラム中では DELETE “AFILE. DT/1”。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。
 ・バブル・カセット中のファイル消

去もできる。

PSP PURGE # “FDi”, “AFILE”

- ・ファイル削除の後、ファイル領域を前につめる。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・ドライブ番号も付けられる。
 KILL “2: DATA”

MLT ・書き込み禁止として、SAVE 時に SAVE “AFILE”, P とした場合ファイル保護はされるが、消去することもできる。
 ・SAVE 参照。

HC KILL “B: TEST”
 ・B はドライブ番号 (A, B, C, D)
 ・ほかは本文と同じ。

FP11 ・ファイル名の前にデバイス名を付ける。省略するとドライブ 0 を指定しないことになる。
 ・ディスク以外に RAM, ROM パックで使用可能。

SMC ERA “AFILE”
 ・機能は本文と同じ。

MZ35 ・本文と同じ。なおディスク上のファイルをすべて消去した時は、KILLALL A0。

X1Hu ・CAS : に対しては実行できない。

MB16 ・MLT と同じ。エクステンションの省略はできない。

IBM55 ・特徴 2 はない。

ファイルの消去 KILL

キル

<p>・エクステンションは省略できない。</p> <p>PC100 ・本文と同じ，なお，ファイル名にはワイルド・カードも使える。</p> <p>・書き込み禁止属性でも消せる。</p>			
<p>MSX —</p>			

ファイル属性の設定と解除 SET

セット

形式: SET 1, "R"

機能: 指定したドライブ、バッファ、ファイルに属性を設定、または解除を行う。

- 特徴: 1 一つ目の引数は、ドライブ番号、# ファイル番号、または "ファイル名" である。
- 2 二つ目の引数は、どの属性を設定するかを決める文字である。属性文字とその意味は次のとおり。
- (a) "R" ... リード・アプタ・ライト (書き込み確認)。
 (b) "P" ... ライト・プロテクト (書き込み禁止)。
 (c) ナル・ストリングまたは "R" "P" 以外の文字 ... 属性を解除する。
- 3 ドライブを指定した場合、そのドライブの中のディスクに対して属性が設定される。
- 4 ファイル名を指定した場合、そのファイルのみに属性が設定され、同じディスク中のほかのファイルには影響しない。
- 5 ファイル番号を指定した場合、以後そのファイルへの出力に対して同じ属性が有効となる。
- 6 "P" を設定すると、PRINT#, PUT, WRITE# などの書き込み動作が禁止され、また、ファイルの KILL, RENAME もできないので、ファイルを保護するのに便利である。
- 7 "R" を設定すると、書き込みを行った直後に読み出しを行えるため、正しく書き込みが行われたか確認ができ便利である。

プログラム例: "afile" に属性を設定し、それを表示する。

```

10 SET "afile", "R"
20 PRINT "attribute="; ATTR$( "afile" )
30 SET "afile", "A"
40 PRINT "attribute="; ATTR$( "afile" )
50 SET "afile", "P"
60 PRINT "attribute="; ATTR$( "afile" )
70 END

```

実行例

```

attribute=R
attribute=
attribute= P

```

参照: ATTR\$

APL LOCK "afile", S1, D2, U3

- ・スロット 1 番で、ドライブ 2 番、ボリューム 3 番のディスク中の "afile" を保護する。
- ・ロックされたファイルは、変更または削除ができない。
- ・カタログ表示の時に、ロックされたファイルには、(*) が付けられる。
- ・解除には、UNLOCK 文を使う。

TRS: ①ATTRIB_ "afile" _
 (param, param)
 ②VERIFY_ ON

- ・①は、"afile" のパスワード、あるいはパスワードによるプロテクト・レベルの変更を行う。
- ・param の内容 (繰り返し、省略可)
 I: ディレクトリ表示にされない。
- ACC= "name1": アクセス・パスワードを "name1" とする。
- UPD= "name2": アップ・デット・パスワードを "name2" とする。
- PROT= level: アクセス・プロテクト・レベルを level にする。
- ・level の内容
 KILL: 制限なし。
 RENAME: ファイル名の変更、書き込み、読み出し、実行は可能。
- WRITE: 書き込み、読み出し、実行は可能。
- READ: 読み出し、実行は可能。
- EXEC: 実行のみ可能。
- ・②は、リード・アプタ・ライトを設定する。解除するには、VERIFY OFF。

CPM ・本文と同じ。

M223 CHATR_ n: "AFILE"/P
 ・n はドライブ #1, 2, 3 の指定で、

省略時はドライブ #0。

- ・P は追加しようとする属性である。
- P: パーマネント・ファイル。
- A: 属性変更禁止。
- R: 読み込み禁止。
- W: 書き込み禁止。
- ・これらは、CLI 指令で行う。

MZK LOCK FD1@2, "afile"

- ・ドライブ 1 番、ボリューム 2 番のディスク上の "afile" を保護する。
- ・ディレクトリ表示の時に、ロックされたファイルには、(*) が付けられる。
- ・解除は UNLOCK 文を使用する。

CBM —

PC8 ・本文参照。

TRS_{II} ・TRS_I と同じ。

PC3 LOCK "afile, BCD"

- ・ボリューム "BCD" のディスク上の "afile" を保護する。
- ・あとは APL と同じ。

IF8₂ ・本文と同じ。レベル₃ —

C180 ・OPEN# 文でファイル使用を宣言する時に属性を設定する。

- ・CLI の SETATR でファイル属性の変更可。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。ただし、
 "RP", "PR" の組み合わせも可能。

FM8 —

ファイル属性の設定と解除 SET

セット

PSP —	MSX —		
PC88 ・本文と同じ。			
N52 ・属性の設定は“P”のみである。			
PC6 —			
MLT ・該当なし。似たものに保護ファイルの設定がある。SAVEを参照。			
HC —			
FP11 SET #1, “R” ・指定したバッファ、ファイルに属性を設定または解除する。ドライブに対しては不可能。 ・ファイル番号を指定した場合、ファイルをクローズすると属性は無効となる。 ・その他は本文と同じ。			
SMC ・ファイルの属性は、CP/Mモードで指定できる。			
MZ35 LOCK “AFILE” ・指定したファイルは保護され、そのファイルに対する書き込み、削除、ファイル名の変更が禁止される。 UNLOCK “AFILE” ・ファイルの保護が解除される。			
X1Hu ・特徴1（c）はナル・ストリングのみ。 ・特徴3はない。			
MB16 —			
IBM55 —			
PC100 —			

マウント文 MOUNT

マウント

形式: MOUNT 1

┌

省略可

機能: ディスケットへのリード/ライトを可能にする。

- 特徴: 1 指定されたドライブ中のディスクの読み出し、書き込みができるようにする。
- 2 ドライブ番号を省略すると、全部のドライブについて1の動作を行う。
- 3 ディスケットのファイル配置表にもし誤りがあれば、予備の配置表を読み出し、メッセージを出力しマウントする。もし、それにも誤りがあれば“bad allocation error”となりマウントされない。

実行例: ドライブ1から“TEST”をロードするために、ドライブ1をマウントし、その後リムーブする。

```
Ok
load"1:TEST"
Disk not mounted
Ok
mount 1
Ok
load"1:TEST"
Ok

remove
Ok
```

参照: REMOVE, OPEN

APL ・該当なし。ディスクがドライブに入っていて、システムがONならいつでもマウント状態である。

TRS_I —

CPM ・本文と同じ。

M223 ・BASIC モードにはいると、自動的にマウント状態になる。

MZK ・該当なし。マスタ・ディスクをドライブにセットしてFDを実行すれば、マウント状態になる。

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 ・該当なし。ディスクをドライブにセットして、電源を入れれば、マウント状態になる。

IF8₂ ・ファイル IFUTY をロードして、9 (Program End) 以外の数字を入力する。

レベル₃ ・PC3 と同じ。

C180 ・本文と同じ。なお、ブレーク・コマンド MO で代用もできる。ドライブ番号は省略できない。

M243 ・M223 と同じ。

MZB ・該当なし。ドライブ電源をON、マスタ・ディスクをセット、MZ-80Bの電源をONにすれば、マウント状態になる。

BUB —

FM8 ・レベル₃と同じ。

PSP ・該当なし。OA-DISK BASICの実行が始まった時、マウント状態になる。

PC88 ・該当なし。N₈₈ BASIC モードでは自動的に行われる。

N52 ・APL と同じ。

PC6 —

MLT ・PC3と同じ。ただし、ディスクを入れ替えた時にRESETを行わないと、“DISK full or I/O error”が出ることもある。

HC ・該当なし。HC-20のディップ・スイッチ bit4をONにして、ドライブ電源もONにし、次にHC-20の電源をONにするとマウント状態になる。

FP11 ・PC3と同じ。

SMC DISK RESET “B:” ・ディスクを交換した時に、新しいディスクは読み出し専用になるので、書き込みを可能にするため行う。

MZ35 ・OPEN 参照。

X1Hu —

MB16 ・PC3 と同じ。

IBM55 ・PC3 と同じ。

PC100 ・PC88 と同じ。

MSX —

リムーブ文 REMOVE

リムーブ

形式：REMOVE 1

┐

省略可

機能：ディスクをドライブから取り出せる状態にする。

- 特徴：1 引数で指定したドライブについて、ディスクにファイル配置表を書き込み、ディスクへの読み出し、書き込みを終わる。
- 2 引数を省略すると、すべてのドライブについて1の動作を行う。
- 3 ディスクをドライブから取り出す前に、必ずこの命令を実行しなければならない。もし怠るとディスクを壊してしまうことがある。

実行例：リムーブしてからディスクを取り出す。

```
Ok
load"1:TEST"
Disk not mounted
Ok
mount 1
Ok
load"1:TEST"
Ok

remove 1
Ok
```

参照：MOUNT

APL ・該当なし。ファイルのアクセス中でなければ、いつでもディスクをドライブから取り出せる。

TRS_I —

CPM ・本文と同じ。RESETと置き換え可能。

M223 ・該当なし。READY状態でファイルがすべてクローズされている状態が、リムーブ状態である。

MZK ・M223と同じ。

CBM ・M223と同じ。

PC8 ・本文参照。

TRS_{II} ・M223と同じ。

PC3 ・M223と同じ。

IF8₂ —レベル₃ ・M223と同じ。

C180 RELEASE 1
・本文と同じ。ブレーク・コマンドREで代用できる。

M243 ・M223と同じ。

MZB ・M223と同じ。

BUB —

FM8 ・M223と同じ。

PSP —

PC88 ・該当なし。N₈₈ BASICモードで自動的に行われる。

N52 ・M223と同じ。

PC6 —

MLT ・該当なし。ただし、RESET命令で同様のことができるが、必ず行う必要はない。また、書き込み禁止の状態を解除する。

HC RESET "A :"
・交換したディスクの書き込みを可能にする。ただし、RESETはディスクを交換した後に実行しなければならない。

FP11 ・M223と同じ。

SMC —

MZ35 ・M223と同じ。

X1Hu —

MB16 ・M223と同じ。

IBM55 ・M223と同じ。

PC100 ・PC88と同じ。

MSX —

バッファ数設定 MAXFILE

マックス・ファイル

形式: MAXFILE 4

バッファ数

機能: 同時に扱えるファイルの数を指定する。

- 特徴: 1 一つのバッファの大きさは、320 バイト、バッファ数を大きくすると、同時にたくさんのファイルをオープンできるが、それだけメモリを使う。
- 2 この文を使う前、バッファ数は3に設定されている。
- 3 プログラム中に使うときは、最初を書く、この文が実行されると、すべての変数はクリアされるので注意。
- 4 MAXFILE n とすると、ファイル番号は #1, #2, ..., #n までが使える。従って、n が 4 のとき、ファイルの一つしか使わないとしても #5 のファイル番号は使えない。
- 5 プログラム中でも使えるので、必要バッファ数の知識のない使用者を想定した、プログラムを書くときに便利である。

用語: ロジカル・ナンバ データやプログラムを、メモリ・ユニットに出し入れするときの、対象となるユニットを指定するための番号。

実行例: 4つのファイルを同時に OPEN するために、MAX FILE 4 を用いる。

```
>LIST
  10 OPEN "0",4,"AFILE:A1"
  20 CLOSE
  30 END
>RUN
ERROR 220 IN LINE 10
>S MAXFILE 4
>RUN
READY
>
```

参照: OPEN, CLOSE

APL MAXFILES 4

・ただし、これは DOS コマンドである、ファイルは最大16個まで。

TRS_I ・システム起動時に入力する、ファイルは最大15個まで。

CPM ・ファイルは最大15個まで。

M223 ・ファイルは8個内で自由に使える。

MZK ・ファイルは10個で固定、ロジカル・ナンバ¹⁾は1から127。

CBM ・ファイルは10個で固定、ロジカル・ファイル番号は1から255。

PC8 ・TRS_Iと同じ。

TRS_{II} ・TRS_Iと同じ。

PC3 ・本文参照。

IF8₂ ・TRS_Iと同じ。

レベル₃ ・ファイルは最大16個まで、設定値は、ユーティリティ・プログラム CONFIG で変えられる。

C180 ・システム起動時に入力する、ファイルは最大255個まで。

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・TRS_Iと同じ。

FM8 ・TRS_Iと同じ。

PSP ・CBMと同じ。

PC88 ・TRS_Iと同じ。

N52 ・TRS_{II}と同じ。

PC6 —

MLT ・TRS_Iと同じ。

HC ・CPMと同じ。ただし、一つのバッファの大きさは143バイトである。

FP11 MOUNT 4
・電源投入時または NEW ALL 後のバッファ数は0に設定されている。

SMC ・ファイルは0から127内で自由に使える。

MZ35 ・ファイルは最大7個まで。

X1Hu ・ファイルは最大15個まで、また、起動時は1になる。

MB16 ・最大255。

IBM55 ・OPEN 文実行時のメモリの大きさに依存、最大で255。

PC100 ・BASIC の起動時に、/F ;
④として指定する、④は整数で0~15だが、指定しないと3になる。

MSX MAXFILES=4
・この文を使う前、バッファ数は1に設定されている。

ファイルのオープン文 OPEN_{ファイル}

オープン

形式: OPEN "AFILE" FOR INPUT AS
#5

省略可

機能: ファイルをオープンし、ファイル番号¹⁾と結びつける。

特徴: 1 INPUT はモードと呼ばれほかに OUTPUT, APPEND と書くことができる。各モードの意味は次のとおり。

(a)INPUT …… ファイルはすでにあるファイルで、ポインタ²⁾はファイルの初めの位置におかれる。

(b)OUTPUT …… 新たなファイルを作り、ポインタはファイルの初めの位置におかれる。

(c)APPEND …… ファイルはすでにあるファイルで、ポインタはファイルの終わりの位置におかれる。

2 ランダム・ファイルの場合、FOR INPUT は省略できる。

3 ファイルを使う時は、まず OPEN 文を実行しなければならない。

4 システム立ち上がり時に "How many files?" と聞いてくるが、そのとき入力した数がファイル番号の上限の値である。

用語: ¹⁾ファイル番号 複数のファイルを識別するためにつけられる番号のこと。

²⁾ポインタ 領域を定めるための先頭アドレスを示すもの。この場合はどのレコードを現在、対象にしているかを示す。

プログラム例: ファイル "AFILE" を作り、書き込み、読み出す。

```
10 PRINT "make & open AFILE":OPEN "AFILE" FOR OUTPUT AS #5
20 PRINT #5,"TEST PROGRAM"
30 PRINT "close AFILE":CLOSE 5
40 PRINT "open AFILE":OPEN "AFILE" FOR INPUT AS #6
50 INPUT #6,E$
60 PRINT E$
70 PRINT "close AFILE":CLOSE 6
80 END
```

実行例

```
make & open AFILE
close AFILE
open AFILE
TEST PROGRAM
close AFILE
```

参照: CREATE, CLOSE, XOPEN, WOPEN

APL D\$=CHR\$(4):PRINT
D\$:"OPEN AFILE"

TRS₁ OPEN "I", 5, "AFILE"
・APPEND モードはない。
・ランダム・ファイルの場合、"I" を "R" とする。

CPM OPEN "I", #5, "AFILE"
・APPEND モードはない。

M223 OPEN "AFILE" FOR
INPUT AS FILE 5
・APPEND モードはない。
・連続ファイルの場合入出力を同時に扱えるので FOR INPUT はいらない。

MZK ・シーケンシャル・ファイルの場合 WOPEN コマンド、ランダム・ファイルの場合 XOPEN コマンドを用いる。

CBM DOPEN #5, "AFILE"
・APPEND は APPEND コマンドを用いる。

PC8 ・本文参照。

TRS_{II} OPEN "D", 5, "AFILE",
64
・APPEND モードはない。
・ランダム・ファイルはレコード長を指定できる。

PC3 OPEN "I", 5, "AFILE"
・ランダム・ファイルの場合 "I" を "R" とする。

IF8 ・本文と同じ。

レベル₃ OPEN "I", #5, "AFILE"
・APPEND モードはない。

C180 OPEN #5:"AFILE",

INPUT

- ・ファイルを作る時は CREATE 文を用いる。
- ・SCRATCH でバッファ内を消去できる。

M243 ・M223 と同じ。ただし、ファイルにパスワードを付けることができる。また、装置名、ディレクトリ名も指定することができる。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・TRS₁ と同じ。
・バブル・カセット中のファイルもオープンできる (BUB0:)。
・"I" の代わりに "A" とすればアペンドもできる。

PSP OPEN # "FD1",
"AFILE", 5, "I"
・モードは本文のほかに全データをクリアして書き直す N モードがある。

PC88 ・本文と同じ。

N52 ・ランダム・ファイルの場合は "FOR モード" を省略しなければならない。省略された場合で、ファイル名が見つからないときは、指定された名前のファイルが作られる。ポインタはファイルの初めの位置に置かれる。特徴 4 において、ファイル番号の上限は 15 で固定されている。

PC6 ・本文と同じ。

MLT ・次のような使い方もできる。
OPEN モード 2, [#] ファイル番号,

ファイルのオープン文 OPEN ファイル

オープン

<p>ファイル名〔、レコード長〕</p> <ul style="list-style-type: none"> ・モード2は文字式で指定。 ・"I"順編成ファイルに対して、入出力処理を指定。 ・"O"順編成ファイルに対して、出力処理を指定。 ・"R"ランダム・ファイルに対して、入出力処理を指定。 ・レコード長は省略可。 	<ul style="list-style-type: none"> ・1番目の引数は、 I：シーケンシャル・ファイルの読み込み O：シーケンシャル・ファイルへの書き出し A：すでにあるファイルへ追加 R：ランダム・ファイル ・デバイス名は、 KEY：キーボード CRT：ディスプレイ・テレビ SCR：ディスプレイ・テレビ LPT：プリンタ CAS：カセット 0：～3：フロッピー・ディスク MEM：グラフィック・メモリ EMM0：～EMM9：外部メモリ ・A、RはCAS：には使えない。 	<p>COM1：RS-232Cポート KYBD：キーボード LPT1：プリンタ SCRN：スクリーン</p> <hr/> <p>MSX OPEN "デバイス名：ファイル名" FOR モード AS ファイル番号</p> <ul style="list-style-type: none"> ・デバイスには、次のものがある。 CAS：カセット CRT：CRTディスプレイ GRP：グラフィック・スクリーン LPT：プリンタ ・ランダム・ファイルはない。 	
<p>HC ・レベル3と同じ。Iモードは入力指定。Oモードは出力指定である。</p> <p>FPI1 ・APPEND モードはない。</p> <ul style="list-style-type: none"> ・特徴4はない。MOUNT文を実行する。 	<p>MB16 ・MLTと同じ。またファイル番号の後に、LEN=レコード長、としてレコード長を加えられる。</p> <ul style="list-style-type: none"> ・デバイス名は、 KYBD：キーボード SCRN：スクリーン LPTn：プリンタ、nは1, 2, 3 A：～D：ミニフロッピー・ディスク COMn：RS-232Cポート、nは、1, 2, 3, 4。 		
<p>SMC OPEN #5, "DSK：AFILE", 256</p> <ul style="list-style-type: none"> ・ランダム・ファイルをオープンする。 ・ROMバックにも使用できる。 ・最後の引数はレコード長で、1から16384までの範囲。 <p>OPEN/I #5, "DSK：AFILE" OPEN/O #5, "DSK：AFILE" OPEN/R #5, "DSK：AFILE" OPEN/S #5, "DSK：AFILE"</p> <ul style="list-style-type: none"> ・/Iは、ASCII型式でファイルされているシーケンシャル・ファイルからデータを読み込む。 ・/Oは、ASCII型式でデータをシーケンシャル・ファイルに書き込む。 ・/Rは、内部表現の型式でファイルされているシーケンシャル・ファイルからデータを読み込む。 ・/Sは、内部表現の型式でデータをシーケンシャル・ファイルに書き込む。 ・ディスク以外にも使用できる。 	<p>IBM55 ・特徴2, 4がなくランダム・ファイルの場合、FORモードは省略できる。</p> <ul style="list-style-type: none"> ・MB16と同じ使い方ができる。 ・ファイル番号は1から255まで。 ・デバイス名はMB16と同じであるが、ディスクはDでなく、通信ポート番号は1または2である。 		
<p>MZ35 ・PC3と同じ。</p> <p>X1Hu OPEN "I", #1, "AFILE"</p>	<p>PC100 ・"How many files"の指定はないが、同様の指定がある。</p> <ul style="list-style-type: none"> ・デバイス名は、 A：またはB：フロッピー・ディスク C：RAMディスク D：またはE：ハードディスク 		

ファイル・クローズ文 CLOSE

クローズ

形式: CLOSE #5, #6

```

  ┌ ┐
  └ ┘

```

*省略・繰り返し可

機能: ファイルをクローズし、ファイル番号と切り離す。

特徴: 1 ファイル番号を省略すると、すべてのファイルをクローズする。
 2 ファイルの使用を終える時には、必ず CLOSE 文を実行しなければならない。

プログラム例: ファイル "AFILE" を作り、書き込み、読み出す。

```

10 PRINT "make & open AFILE":OPEN "AFILE" FOR OUTPUT AS #5
20 PRINT #5,"TEST PROGRAM"
30 PRINT "close AFILE":CLOSE 5
40 PRINT "open AFILE":OPEN "AFILE" FOR INPUT AS #6
50 INPUT #6,E$
60 PRINT E$
70 PRINT "close AFILE": CLOSE 6
80 END

```

実行例

```

make & open AFILE
close AFILE
open AFILE
TEST PROGRAM
close AFILE

```

参照: OPEN

APL D\$=CHR\$(4):
PRINT D\$;"CLOSE AFILE"

TRS₁ CLOSE 1, 2
 ・END, NEW, RUN, MERGE,
 EDIT, CLEAR 命令は自動的に
 すべてのファイルをクローズする。

CPM ・本文と同じ。

M223 CLOSE 1

MZK CLOSE #1

CBM DCLOSE #1

PC8 ・本文参照。
 ・END, NEW 命令は自動的にすべ
 てのファイルをクローズする。

TRS_{II} ・TRS₁と同じ。

PC3 CLOSE 1, 2
 ・LOAD, CHAIN, MERGE, NEW
 命令は自動的にすべてのファイル
 をクローズする。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 CLOSE #1

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・本文と同じ。

FM8 CLOSE #1, #2
 ・END 命令は自動的にすべてのフ
 ァイルをクローズする。

PSP CLOSE #1, 2

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・本文と同じ。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。なお、
 NEW, RUN, LOAD のときもす
 べてのファイルは閉じられる。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。なお、
 END, NEW, RESET, SYSTEM,
 R オプションなしの RUN 命令は、
 すべてのファイルをクローズする。
 ・ほかに、ディスケット・ファイル
 をクローズし、システム・バッフ
 ァを消去する RESET 命令がある。

PC100 ・IBM55と同じ。

MSX ・本文と同じ。

フィールド文 FIELD

フィールド

形式: FIELD #2, 50 AS E1\$, 20 AS E2\$

*省略・繰り返し可

機能: ランダム・ファイル・バッファ中のフィールド構成を決める。

- 特徴: 1 ランダム・アクセス・ファイルの固定長¹⁾レコードの256バイトを、フィールドごとに分割するように指定する。
 2 初めの引数は1~15までの整数で、分割するファイル番号を指定する(関数DSKI\$を使うときは0を指定する)。
 3 コマ(,)の後に各フィールドの長さ(バイト)を書く。
 4 フィールドとして指定した文字変数を、LET文の左辺、INPUT文、READ文などで使ってはならない。
 5 例として“形式”で示したレコード構成は、以下のようになる。

フィールド	フィールド	あき
50バイト	20バイト	(186バイト)

レコード (256バイト)

- 6 ランダム・ファイルを使う前に定義しなければならない。

用語: ¹⁾固定長 一つのレコードの長さが限定され、一定であること。☐可変長。
²⁾RAMファイル メモリ上に作れるファイル。電源を切っても、データが保存されるものもある。複数のプログラム間でデータを共用することもできる。

プログラム例: KYU-fに名前と預金残高の領域をフィールド文で、それぞれ15バイト、5バイトをとり、それらをランダム・バッファに書き込み、まとめて出力する。その後、ファイル名を“SHIN-f”に変える (NAMEを参照)。

実行例

```

** YOKIN ZANDAKA **
miyazaki      5000円
A.nakamura    8900円
nakanishi     10000円
noda          10000円
nishida       7500円
horiuchi      15000円
suzuki        6000円
Y.nakamura    10000円

```

参照: LSET, RSET, PUT, GET

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。ただし、ランダム・ファイルのバッファ長は128バイト。

M223 RECORDSIZ #1, E\$(20), B1(0)
 ・OPEN文、入出力文の前に実行する必要がある。配列数値変数も使用できる(単精度数のとき自動的に4バイト長の領域をとる)。ほかは本文と同じ。ただし、物理的には256バイトだが、論理的には任意長とれる。

MZK ・該当なし。各数値変数、文字変数の領域は16バイトに固定されている。

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 RFORMAT #1, A%, E\$
 ・数値変数、文字変数双方とも変数領域が割り当てられる。領域の長さの指定の必要はない。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。ただし、1レコード長は128バイト。

C180 DIM E1\$ *50
 DIM A(1)
 ・上のように配列宣言でバッファを指定する。
 ・RECDEFで階層的に文字変数に文字数を割り当てることができる。

M243 ・M223と同じ。

MZB ・該当なし。各数値変数、文字変数の領域は32バイトに固定されている。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP BUILD # “FD2”, “ファイル名”, “IS”, 127, 1, 14, 200, 2000, “XYZ”
 ・FD2 (フロッピー第2装置), 127(レコード長), 1(キーの位置), 14(キーのバイト数), 200(ファイル・サイズ), 2000 [エラー処理文番号(省略可)], XYZ(パスワード(省略可))。
 ・ファイル数によらず1Kバイトごとにディスク上に領域がとられる。

PC88 ・本文と同じ。ただし、カセットに対しても同じように使うことができる。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・1レコード長は128バイト。
 ・特徴1, 5は不可。
 ・フィールド長さ(50と20)は文字数を表す。
 ・RAMファイル²⁾の大きさ定義はDEFFIL命令を用いる。

FP11 ・本文と同じ。

SMC RECORD #2, A, N,
 RJUST 20 AS F\$(2)

フィールド文 FIELD

フィールド

<ul style="list-style-type: none"> ・レコード長はOPEN命令で指定できる。 ・ファイル番号は 0 から 127. ・整数型は 2 バイト、実数型は 8 バイトに設定され、文字型は LJUST, RJUST で指定する。 ・配列変数は RECORD 文中で配列宣言される。 			
MZ35 ・ PC3 と同じ。			
X1Hu ・ 本文と同じ。			
MB16 ・ 特徴 2 はない。 ・ レコード長は 1 から 32767 の範囲。			
IBM55 ・ MB16 と同じ。			
PC100 ・ 本文と同じ。			
MSX —			

ディレクトリ出力 FILES

ファイルズ

<p>形式：FILES 1 └ 省略化</p> <p>機能：ディスク中の全ファイルを CRT に出力する。</p> <p>特徴：1 ドライブ番号を省略すると、1 を指定したことになる。 2 表示はファイル名、拡張子、ファイルの大きさの順。 3 ファイル名を忘れた時などに使うと便利である。</p> <p>用語：1) ダンプ 記憶装置の中に入っている内容を、画面表示したり印刷すること。</p>	<p>APL CATALOG 1 ・機能は本文と同じ。</p> <p>TRS₁ ・DOS モードで DIR コマンドを実行する。</p> <p>CPM FILES "A:*.*)" ・表示するファイル名を制限できる。</p> <p>M223 ・CLI モードで LIST / 1. 省略すると 0 を指定したことになる。 ・ファイルの大きさは 256 バイト単位。 ・作成日付け、ファイルの型など詳しい情報も同時に表示される。</p> <p>MZK DIR 1 ・番号を省略すると 1 を指定したことになる。</p> <p>CBM DIRECTORY D1 ・機能は本文と同じ。</p>	<p>M243 ・M223 と同じ。</p> <p>MZB DIR FD1 ・FD1 を省略すると一番最近 DIR コマンドを実行したドライブを指定したことになる。</p> <p>BUB ・本文と同じ。</p> <p>FM8 FILES "1:" ・デバイス名をそれぞれ "CAS0:"、"BUB0:" とすると、カセット中およびバブル・カセット中の全ファイル名を出力する。 ・ドライブ番号の省略時は 0。</p> <p>PSP CATALOG 1 ・機能は本文と同じ。</p> <p>PC88 ・本文と同じ。</p>
<p>実行例：ディスキットの中のファイルの一覧を見る。</p> <pre>files akerfh. 1 word .ora 9 word .tad 5 word .3 5 a\$data dat 1 AFILE 1 test1 1</pre> <p>参照：LFILES</p>	<p>PC8 ・本文参照。 ・ファイルの大きさはクラスタ (2K バイト) 単位。</p> <p>TRS_{II} ・TRS_I と同じ。</p> <p>PC3 CAT A0 ・チャンネル記号 (A)、ドライブ番号 0 を与える。</p> <p>IF8₂ ・本文と同じ。</p> <p>レベル₃ FILES "0:" ・デバイス名を CAS0 とするとカセット中の全ファイル名を出力する。</p> <p>C180 ・CLI モードで DIRECT0. ・FILE(A) は A 番のファイル状態を表示する。 ・EXFNO はエラーが発生したファイル番号を与える関数。</p>	<p>N52 ・ドライブ番号を省略すると、0 を指定したことになる。 ・表示は、ファイル名、状態、レコード件数の順。 ・状態は 3 桁で表示され、ファイルの種類、オープン中かどうか、属性が表示される。</p> <p>PC6 ・拡張子なし。ほかは本文と同じ。</p> <p>MLT ・CPM と同じ。ただし、ドライブを省略すると A を指定したことになる。</p> <p>HC ・レベル₃ と同じ。ただし、ドライブ番号は AorB (or C or D) ドライブである。</p> <p>FP11 FILES "0:" ・デバイス名を省略すると、ドライブ 0 が指定される。 ・デバイスは、ディスクと RAM。</p>

ディレクトリ出力 FILES

ファイルズ

<p>ROM バックのみ有効。 ・ファイルの属性も表示される。</p> <hr/> <p>SMC DIR "A : * . *" ・表示するファイル名を制限できる。 ・ファイルの大きさはキロ・バイト単位。 ・USER 文で指定されたユーザ領域中のファイルを表示する。 ・DIR 単独では、現在使用中のファイルすべてを表示する。 ・これに関連して、ASCII セーブされたファイルをダンプ¹⁾する TYPE 命令がある。</p> <hr/> <p>MZ35 ・PC3 と同じ。</p> <hr/> <p>X1Hu FILES "0 :" ・ファイル・ディスクリプタを省略すると、DEVICE で指定したファイル名になる。 ・表示はファイル型式、ファイル名、作成日時の順。 DEVICE "1 :" ・ファイル・ディスクリプタ省略時のデバイスを変更する。</p> <hr/> <p>MB16 ・CPM と同じ。また、疑問符 (?) を含むと、ファイル名やエクステンション中の 1 文字の代わりをする。</p> <hr/> <p>IBM55 ・MB16 と同じ。</p> <hr/> <p>PC100 ・本文と同じ。ただし、ドライブ名、パス名を省略すると、カレント・ディレクトリが選択される。 ・カレント・ディレクトリの、 変更は CHDIR 登録は MKDIR 削除は RMDIR により実行される。 ・もっと細かく MS-DOS が BASIC 用に用意した環境エリアのパラメ</p>	<p>ータを変更するのに ENVIRON 文、ENVIRON \$ 関数を用いる。</p> <hr/> <p>MSX —</p> <hr/>		
--	---	--	--

ディレクトリのプリンタ出力 LFILES

エル・ファイルズ

形式：LFILES 1
 省略可

機能：ディスク中の全ファイル名をプリンタに出力する。

特徴：1 プリンタに出力すること以外は FILES と同じ。
 2 ディスケットの整理に使うと便利である。

実行例：ディスクットの中のファイルの一覧をプリンタに出力する。

lfiles

```
akgrph.      1      word .org 9
word .3      5      a#data dat 1
testi        1
```

参照：FILES

APL —

TRS_I —

CPM ・DOS モードで (CTL) -
 [P] の後 DIR.

M223 ・CLI モードで LIST [L]
 SOUT/3. LIST のみでは CRT
 へ表示.

MZK DIR/P
 ・機能は本文と同じ。

CBM ・CMD4 と打ってプリンタ
 をオープンし、DIRECTORY コ
 マンドを実行する。

PC8 ・本文参照。

TRS_{II} ・TRS_I と同じ。

PC3 CAT PRINT A0
 ・機能は本文と同じ。

IF8₂ ・本文と同じ。

レベル₃ —

C180 ・CLI モードで DIRECT
 0, \$LP.

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 FILES "1:", L
 ・機能は本文と同じ。

PSP CLIST
 ・機能は本文と同じ。

PC88 ・本文と同じ。

N52 —

PC6 ・拡張子はない。

MLT —

HC ・FILES と COPY 参照。

FP11 LFILES "0:"
 ・機能は本文と同じ。
 ・デバイス名を省略すると、ドライ
 ブ 0 が指定される。

SMC ・(CTR)-[P] の後 DIR.

MZ35 CAT A0*/
 ・チャネル記号 (A), ドライブ番号
 (0) を指定する。

X1Hu ・FILES を参照。

MB16 ・CPM と同じ。

IBM55 ・FILES のあと、ハード
 ・コピーをとる。

PC100 ・Shell コマンドを使い、
 DIR を使って同じことができる。

MSX —

フォーマットING FORMAT

フォーマット

形式: FFORMAT 1
 省略可

機能: ディスケットをフォーマットING¹⁾する。

- 特徴: 1 ドライブ番号を省略すると1を指定したことになる。
 2 市販のディスクは、まずフォーマットINGしなければなら
 ない。
 3 このコマンドを実行すると、ディスクの前の内容は失われ
 る。
 4 通常はメーカーで用意した format プログラムを用い、この命令は
 使用しない。

用語: ¹⁾フォーマットING 市販のフロッピー・ディスクを使えるようにするため初期化
 を行うこと。イニシャライズともいう。
²⁾ユーティリティ・プログラム OS (Operating System)に含まれる。プログラム
 で、ユーザにいろいろな便宜を与えてくれる。

実行例: ユーティリティ・プログラム format の実行例。

```
run"format"
create system disk(y/n)? n
mount new disk on drive 2
sure(y/n)? y
complete
Ok■
```

参照:

APL ・ greeting program を作り
 そのファイル名をたとえば
 HELLO とする。そして、INIT
 HELLO と打つ。

TRS₁ ・ DOS モードで FORMAT
 プログラムを実行する。

CPM ・ TRS₁ と同じ。

M223 ・ CLI モードで INIT/0。
 ・ イニシャライズされたディスケッ
 トはデータ・ディスクとして使え
 る。

MZK ・ ユーティリティ・プログ
 ラム²⁾ "S-DISKETTE INIT" を
 実行する。

CBM HEADER "NEWDISK",
 D1, 199
 ・ ディスク名、ID 番号を与える。

PC8 ・ BASIC で使うには
 format プログラムを実行する。
 ・ format プログラムを実行するとシ
 ステム・ディスクを作るかどうか
 聞いてくる。システム・ディス
 クを作らない場合、データ・ディス
 クが作られる。

TRS_{II} ・ TRS₁ と同じ。

PC3 INIT
 ・ 上記コマンド入力後 VOLUME
 NAME? と聞いてくるので、ボ
 リューム名を入力する。
 ・ ボリューム名の変更は、LPRINT
 A1, "VOLNAME" による。
 ・ ボリューム名を知りたいときは、
 LINPUT A1, X\$ でわかる。

IF8₂ ・ ユーティリティ・プログ
 ラム "IFUTY" の 1 と 2 を実行す
 る。

レベル₃ ・ FORMAT プログラムを
 実行し、その後 DSKINI コマンド
 を実行する。

C180 ・ CLI の VINIT コマンド
 で行う。

M243 ・ M223 と同じ。

MZB ・ ユーティリティ・プログ
 ラムで "I" と "S" コマンドを
 実行する。

BUB ・ マニュアル中のフォー
 マット・プログラムを実行する。

FM8 ・ ユーティリティ・プログ
 ラム "SYSDSK" を実行し、そ
 の後 DSKINI コマンドを実行す
 る。

PSP —

PC88 ・ 標準フロッピーはあらかじ
 めイニシャライズされている。ミ
 ニ・フロッピーはプログラム "format.
 n88" を実行させる。

N52 ・ ドライブ番号を省略する
 と 0 を指定したことになる。

PC6 —

MLT ・ ユーティリティ・プログ
 ラムの "FDD INITIALIZE" を
 実行する。

HC FRMAT "A:"
 ・ ドライブ番号は A, B, C, D のい
 ずれか指定する。
 ・ システムのコピーには SYSGEN
 コマンドを用いる。

FP11 FORMAT "1"
 ・ FORMAT の後にデバイス名を付

フォーマット **FORMAT**

フォーマット

ける。デバイス名の両端には(") を付ける。 ・デバイス名を省略するとドライブ 0が指定される。 ・ディスク以外にRAM、ROMパッ クのフォーマットが可能。 ・確認をしないので使用には注 意が必要。			
SMC ・TRS ₁ と同じ。			
MZ35 ・PC3 と同じ。			
X1Hu ・ユーティリティ・プログ ラムのイニシャライズを用いる。			
MB16 ・DOS レベルでユーティ リティ・プログラム "FORMAT" がある。			
IBM55 ・MB16 と同じ。			
PC100 ・MB16 と同じ。			
MSX —			

ディレクトリの作成 CREATE

フリエート

形式: CREATE "AFILE. DT/1",

↑
必要

省略可

種別 ドライブ No

CONTINUOUS (100, 5) *

レコード長 レコード数

機能: ディレクトリ¹⁾を作成し、ファイル名を登録する。特徴: 1 (a) CONTINUOUS は連続ファイル²⁾であることを指定している。(b) 連続した物理レコード³⁾を確保する。

ディレクトリにファイル名とレコード長を登録する。

2 (a) CONTINUOUS 以降を LINKED とすれば、リンクド・ファイル⁴⁾指定となる。

(b) ディレクトリにファイル名とレコード長 (256 バイト) を登録する。

用語: ¹⁾ディレクトリ フロッピー・ディスクに格納されているすべてのファイルの名前、属性、格納されている場所の先頭を示す表。²⁾連続ファイル [SORD, C180] 物理的に連続した領域に作られるファイル、ファイル内のデータの読み書きは任意にできる。[SORD] ☐ シーケンシャル・ファイル ☐ [C180] ☐ リンクド・ファイル ☐ 論理レコード³⁾物理レコード ある記憶媒体に記憶されている、隣接したデータの集合、通常、その全体がハード的に入出力転送される。⁴⁾リンクド・ファイル [C180] 必ずしも物理的に連続しない領域に作られるファイル、1セクタのデータとして使われる領域は 256 バイトであり、ファイル内のデータの読み書きは逐次的にしかできない。 ☐ 連続ファイル

プログラム例: 実行例は RESTORE を参照。

```

100 DIM B(1)
110 CREATE "AFILE/1",CONTINUOUS(10,9)
120 OPEN #1:"AFILE/1",INPUT,FIXED(10)
130 AT EOF #1:240
140 FOR I=1 TO 10
150   B(1)=I*3.14159
160   PUT #1:B
170 NEXT I
180 RESTORE #1
190 N=1
200 GET #1:B
210 PRINT "チョッカイ=";N,"エンジウ=";B(1)
220 N=N+1
230 GOTO 200
240 PRINT "END"
250 END

```

参照: OPEN

ディレクトリの作成 CREATE

APL	・ OPEN 参照。	"AFILE", "RF", 100, , , 5で同様のことができる。"FD1"はドライブ番号で、FD1~FD4が使える。"RF"は直編成ファイルを示す。ここを"SV"とすれば順編成ファイルを示し、"ISF"とすれば、索引順編成ファイルであることを示す。	
TRS ₁	・ OPEN 参照。		
CPM	・ OPEN 参照。		
M223	・ 該当なし。ただし、CLI コマンドで CREATE 文を実行すれば同様のことができる。OPEN 参照。		
MZK	・ OPEN 参照。	PC88	・ OPEN 参照。
CBM	・ OPEN 参照。	N52	・ OPEN 参照。
PC8	・ OPEN 参照。	PC6	・ OPEN 参照。
TRS _{II}	・ 該当なし。ただし、DOS コマンドで CREATE 文を実行すれば同様のことができる。OPEN 参照。	MLT	・ OPEN 参照。
PC3	・ CREATE "AFILE: A1", 5, 100で同様のことができる。 ・ レコード長を省略すると256 バイトとなる。レコード数も省略すると、レコード数8、レコード長 256 バイトとなる。	HC	・ OPEN 参照。
IF8 ₂	・ OPEN 参照。	FP11	・ OPEN 参照。
レベル ₃	・ OPEN 参照。	SMC	・ OPEN 参照。
C180	・ 本文参照。	MZ35	・ OPEN 参照。
M243	・ M223 と同じ。	X1Hu	・ OPEN 参照。
MZB	・ OPEN 参照。	MB16	・ OPEN 参照。
BUB	・ OPEN 参照。	IBM55	・ OPEN 参照。
FM8	・ OPEN 参照。	PC100	・ OPEN 参照。
PSP	・ BUILD # "FD1",	MSX	—

レコード長の変更 MARGIN #n

マージン #n

形式: MARGIN #5:128

↑ — —
必要 ファイル 長さ指定
 ル番号

機能: OPEN #n で指定したレコードの長さを変更する。

- 特徴:
- 1 ディレクトリのレコード長も変更する。
 - 2 ファイル番号、長さ指定は算術式が使える。
 - 3 入出力装置による特徴。
 - (a) コンソール (ファイル番号を 0 とすれば、コンソールが指定される) の場合
CRT 出力の右端の指定 ($1 \leq \text{長さ指定} \leq 80$)。
 - (b) プリンタの場合
プリンタの 1 行の長さの指定。
 - (c) フロッピー・ディスクの場合。
OPEN #n で指定したレコード長の変更。
 - (d) シリアル・インターフェースの場合
OPEN #n で指定したレコード長の変更。

プログラム例: プリンタの 1 行の文字数を変える。

```
10 DIM E$*30
20 OPEN #6:$LP(136)
30 E$="ABCDEFGH IJKLMNOPQRSTUVWXYZ"
40 OUTPUT #6:E$
50 MARGIN #6:20
60 OUTPUT #6:E$
70 MARGIN #6:136
80 CLOSE #6
90 END
```

実行例

```
ABCDEFGHIJKLMN O PQRSTUVWXYZ
ABCDEFGHIJKLMN O PQRST
UVWXYZ
```

参照:

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 —

IF8₂ —レベル₃ —

C180 ・本文参照。

M243 —

MZB —

BUB ・該当なし。ただし、
WIDTH LPRINT 80 (文字数)
でプリンタの 1 行の長さを指定で
きる。

FM8 —

PSP —

PC88 ・BUB と同じ。

N52 —

PC6 —

MLT —

HC WIDTH "LPT0:", 20

・プリンタの一行印字数の最大を設
定する。桁数は 1 から 255 ままで、
255 で印字幅による自動改行を行
わない。

・"COM0:" で RS-232C ポートの
桁数設定もできる。

・DEFFIL 20 , 200
レコード長さ 相対番地

使用する RAM ファイルの相対番
地。1 レコード長さを定義する。
・相対番地とは CLEAR 文で指定し
た RAM ファイル・エリアの開始
番地からのバイト数である。

FP11 ・該当なし。ただし、
LWIDTH M (M は文字数) でプリ
ンタの一行の長さを指定できる。
文字数は 80 または 132。

SMC ・OPEN 文でレコード長を
変更することができる。

MZ35 —

X1Hu —

MB16 WIDTH #1, 200

・ファイル番号に対応するデバイ
スの出力幅を直ちに設定。デバイス
はプリンタ、スクリーン、コミュ
ニケーション・ポート。

WIDTH "LPT1:", 80

・設定の変更はすぐにせず、次の
OPEN 文実行後。ただし LIST,
LLIST でデバイスがプリンタの
場合は自動的に OPEN される。

IBM55 ・MB16 と同じ。

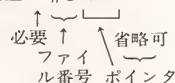
PC100 ・BUB と同じ。

MSX —

ファイル・ポインタの移動 RESTORE #n

リストア #n

形式: RESTORE #9:4



機能: 指定した位置へファイル・ポインタを移す。

- 特徴: 1 ポインタ・アドレスを省略すると、ポインタ・アドレスは0となる。
 2 ポインタ・アドレスを省略すれば、可変長¹⁾レコード・ファイルにも使用できる。

用語: ¹⁾可変長 ユーザーが1レコードを任意の長さに指定できること。□固定長

実行例: プログラムは CREATE を参照。

```

チャックイン 1      インシュウ 3.14159
チャックイン 2      インシュウ 6.24159
チャックイン 3      インシュウ 9.44159
チャックイン 4      インシュウ 12.4159
チャックイン 5      インシュウ 15.4159
チャックイン 6      インシュウ 18.4159
チャックイン 7      インシュウ 21.4159
チャックイン 8      インシュウ 25.4159
チャックイン 9      インシュウ 28.4159
チャックイン 10     インシュウ 31.4159
END
  
```

参照: GET, PUT

APL —

TRS₁ ・GET #9, 4 とすれば、ポインタを移動させ、データを入力できる。
 ・PUT #9, 4 とすれば、ポインタを移動させ、データを出力できる。

CPM ・TRS₁ と同じ。

M223 GET #9 RECORD 4
 ・ポインタを移動させ、データを入力する。
 PUT #9 RECORD 4
 ・ポインタを移動させ、データを出力する。

MZK INPUT #9 (4), E\$
 ・ポインタを移動し、データを入力する。
 PRINT #9 (4), E\$
 ・ポインタを移動し、データを出力する。

CBM RECORD #9, 4
 ・機能は本文と同じ。

PC8 ・TRS₁ と同じ。TRS₁₁ ・TRS₁ と同じ。PC3 ・TRS₁ と同じ。IF8₂ ・TRS₁ と同じ。レベル₃ ・TRS₁ と同じ。

C180 ・本文参照。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・TRS₁ と同じ。FM8 ・TRS₁ と同じ。

PSP —

PC88 ・TRS₁ と同じ。N52 ・TRS₁ と同じ。

PC6 —

MLT ・TRS₁ と同じ。

HC —

FP11 ・TRS₁ と同じ。SMC ・TRS₁ と同じ。

MZ35 ・MZK と同じ。

X1Hu ・TRS₁ と同じ。MB16 ・TRS₁ と同じ。IBM55 ・TRS₁ と同じ。PC100 ・TRS₁ と同じ。

MSX —

VIDEO-RAM データのセーブとロード VSAVE, VLOAD

ビデオ・セーブ, ビデオ・ロード

形式：① VSAVE * "AFILE", M1, M2

省略化

② VLOAD "AFILE"

機能：①CRT上の(M1+1)行目から(M2+1)行目までの画面分をディスクへ格納する。

②ディスクへ格納されていた画面情報"AFILE"を、CRT上へ再現する。

特徴：1(a)PC-3200(S)では、(*)を付けることで、罫線、カラー、文字状態(プリンク¹⁾、リバース²⁾)のみの格納に制限できる。

(b)M1, M2を省略すると、画面全体の情報が格納される。

2 VSAVEで格納された行以外は、影響を受けない。

用語：¹⁾プリンク 画面上で、使用者へ注意をうながすために、文字を点滅させること。
²⁾リバース 色が逆転すること。たとえば、現在CRT上に出力されている文字が緑色で、バックの色が黒ならば、リバース(反転)すると、文字が黒にバックが緑になる。

プログラム例：CRTの上から3行を(*)10個ずつで埋め、その3行をセーブし、CRTの最上行を(+)10個で埋めた後、今度はディスクからその3行を読み込み、CRT上へ表示する。

```
10 CURSOR 0,0
20 FOR I=1 TO 3
30 FOR J=1 TO 10
```

```
40 DISP "*";
50 NEXT J:DISP
60 NEXT I
```

```
70 VSAVE "AFILE:A1",0,3
```

```
80 CURSOR 0,0
```

```
90 DISP "++++++++++"
```

```
100 CURSOR 0,4
```

```
110 DISP "NOW LOADING!"
```

```
120 VLOAD "AFILE:A1"
```

```
130 END
```

実行例

NOW LOADING!

READY

>

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 ・ 本文参照。

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ①GSAVE "AFILE", X1, Y1, X2, Y2

・ X1, Y1 は左上すみの座標。

・ X2, Y2 は右下すみの座標。

②GLOAD "AFILE", X1, Y1

・ X1, Y1 は開始座標。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

参照：

プログラム実行の交換 SWAP プログラム

スワップ

形式：SWAP FDM1@M2, "AFILE"

ファイル名

機能：ディスク内プログラムとメモリ内プログラムを交換する。

- 特徴：
- 1 M1はドライブ番号、M2はボリューム¹⁾・ナンバ、
 - 2 実行中のテキストをアクティブ・ドライブ中のディスク上にいったん待避させ、次に、ドライブ M1 中のボリューム・ナンバ M2 のディスク上にある BASIC テキスト "AFILE" を、BASIC テキスト・エリアに読み出し、その先頭からプログラム実行を続ける。"AFILE" の実行終了後、今度はもとのテキストを読み出し、SWAP の次のステートメントからプログラムを実行する。
 - 3 各プログラム実行のリンクの際には、変数の値、利用者関数の内容は受け渡される。
 - 4 SWAP されたテキストで、更に SWAP 命令を行えない。
 - 5 独立で作ったプログラムを利用したい時に便利である。

用語： ¹⁾ボリューム 1 巻のテープ、1 個のディスクなどをファイルと区別するために使う。ボリュームの違いを示すために、ボリューム・ナンバを用いる。

プログラム例：メイン・プログラムで?の三角形と五つの*を出力し、次に"TEST1"のプログラムで!で逆三角形を出力し、最後に"TEST END"をメイン・プログラムで出力する。

```
10 J=4
20 FOR I=1 TO J
30 PRINT/P STRING$( "?", I)
40 NEXT I
50 PRINT/P STRING$( " *", J+1)
60 SWAP FD2@2, "TEST1"
70 PRINT/P "TEST END"
80 END
```

実行例

```
?
??
???
????
*****
!!!!
!!!
!!
!
```

TEST END

ドライブ2 番中のボリューム2にある
"TEST 1" のプログラム、

```
10 FOR I=J TO 1 STEP -1
20 PRINT/P STRING$( "! ", I)
30 NEXT I
40 END
```

参照：GOSUB, CHAIN, CALL

APL —

TRS_I —

CPM —

M223 —

MZK ・ 本文参照。

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB ・ 本文と同じ。

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

セクタ入力 DSKI\$

ディスク・インプット\$

形式：DSKI\$ (1, 2, 3)

機能：ディスクの指定セクタ¹⁾の内容を与える関数。

- 特徴：1 指定はドライブ番号、トラック²⁾番号、セクタ番号の順に行う。
 2 内容は文字列として与えられる。
 3 文字列の長さは、セクタのバイト数と同じ。
 4 DSKO\$と逆の働きをする。
 5 FIELD 文で#0のバッファを用意してから使う。
 6 ディスクの内容を強制的に読むのに便利である。

用語：¹⁾セクタ フロッピー・ディスク上の1トラックの16分の1で、256バイトの容量がある（1トラックの10分の1を1セクタとする機種もある。ただし、容量は同じく256バイト）。

²⁾トラック(トラック番号) フロッピー・ディスク上の35の同心円の一つで、各々が16または10の扇形(セクタ)よりできている。

プログラム例：ドライブ2, 1トラック, 1~10セクタへの書き込み、読み出し。

```
10 FIELD #0,10 AS A$
20 FOR I=1 TO 5
30 LSET A$=STR$(I*10)
40 DSKO$(2,1,I)
50 NEXT I
60 FOR I=1 TO 5
70 PRINT LEFT$(DSKI$(2,1,I),10)
80 NEXT I
90 END
```

実行例

```
10
20
30
40
50
```

参照：DSKO\$, GET

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ —

PC3 —

IF8₂ DSKI\$ (1, 0, 2, 3)
 ・2番目の引数は、サイド番号（表=0, 裏=1）である。

レベル₃ ・本文と同じ。

C180 —

M243 —

MZB —

BUB ・IF8₂と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・IF8₂と同じ。

N52 —

PC6 ・本文と同じ。

MLT —

HC ・ディスク・ナンバは

“A:” (AはB, C, Dでも可)と指定する。

FP11 DSKI 0, 0, 0, 1

- ・指定は、ドライブ番号、サイド番号、トラック番号、セクタ番号の順に行う。
- ・内容は0番バッファに読み込まれる。

SMC —

MZ35 —

X1Hu DEVI\$ “0:”, 1, E\$, E2\$

- ・指定は、ファイル・ディレクトリ、レコード番号で256バイト分の文字列データが、128バイトずつ二つの文字変数に入る。
- ・ディレクトリはディスクのほかにかセット、グラフィック・メモリ、外部メモリも使える。

MB16 —

IBM55 —

PC100 DSKI (2, 1, 1, 1, 2, 9/9)

- ・4番目の引数はメディア・タイプ（片面0, 両面1）、5番目はセクタ長、6番目はセクタ数、7番目はトラック番号、ほかは本文と同じ。

MSX —

セクタ出力 DSKO\$

ディスク・アウトプット\$

形式：DSKO\$ 1, 2, 3

機能：フロッピー・ディスクの指定したセクタに直接文字列を出力する。

- 特徴：
- 1 指定はドライブ番号、トラック番号、セクタ番号の順に行う。
 - 2 文字列の最大の長さはセクタのバイト数。
 - 3 DSKI\$ と逆の働きをする。
 - 4 文字列は、0 番ディスク・バッファにあらかじめ書き込んでおく。
 - 5 ディスクの内容を強制的に書き替えるのに便利である。

プログラム例：ドライブ2, 1 トラック, 1~10セクタへの書き込み、読み出し。

```
10 FIELD #0,10 AS A$
20 FOR I=1 TO 5
30 LSET A$=STR$(I*10)
40 DSKO$2,1,I
50 NEXT I
60 FOR I=1 TO 5
70 PRINT LEFT$(DSKI$(2,1,I),10)
80 NEXT I
90 END
```

実行例

```
10
20
30
40
50
```

参照：DSKI\$, PUT

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ DSKO\$ 1, 0, 2, 3
・2 番目の引数はサイド番号（表＝0, 裏＝1）である。

レベル₃ DSKO\$ 1, 2, 3, E\$
・書き込む内容を E\$ で直接指定する。

C180 —

M243 —

MZB —

BUB ・IF8₂ と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・IF8₂ と同じ。

N52 —

PC6 ・本文と同じ。

MLT —

HC DISKO\$ "A: ", 1, 5,
A\$

・レベル₃ と同じ。

FP11 DSKO 0, 0, 0, 1

・指定は、ドライブ番号、サイド番号、トラック番号、セクタ番号の順に行う。
・0 番バッファの内容を出力する。

SMC —

MZ35 —

X1Hu DEVO\$ "0:", 1, E\$,
E2\$
・ファイル・ディスクリプタで指定したデバイスに DEVI\$ と逆のことを行う。

MB16 —

IBM55 —

PC100 ・DSKI\$ 参照。

MSX —

ディスク関数 DSKF

ディスク・フリー

形式: DSKF (1)

機能: ディスクの未使用領域の大きさを示す。

- 特徴:
- 1 指定はドライブ番号で行う。
 - 2 値はクラスタ¹⁾(2Kバイト)単位。
 - 3 大きなファイルを作る前に作れるかどうか確かめるのに便利である。

用語: ¹⁾クラスタ フロッピー・ディスク上のファイルの読み書きの最小単位は1セクタであるが、8セクタをひとまとめにしてクラスタと呼び管理の最小単位としている。

実行例: ファイルを削除して dskf 関数の値の変化を見る。

```
Print dskf(1)
28
Ok
kill "AFILE"
Ok
Print dskf(1)
29
Ok
```

参照: LIST (M223)

APL —

TRS₁ ・ DOS モードで FREE コマンドを実行する。
・ 値はグラニューール (1,2Kバイト) 単位。

CPM ・ DOS モードで STAT コマンドを実行する。
・ 値はバイト単位。

M223 ・ CLI モードで DISK/0。
・ DISK は LIST コマンドに含まれている。
・ 値はブロック (256 バイト) 単位。

MZK —

CBM ・ DIRECTORY コマンドで知ることができる。

PC8 ・ 本文参照。

TRS₁₁ ・ TRS₁ と同じ。

PC3 ・ CAT コマンドで知ることができる。

IF8₂ ・ 本文と同じ。

レベル₃ ・ 本文と同じ。ただし、値はグループ (0.5Kバイト) 単位。

C180 ・ CLI モードで DIRECT コマンドを実行すれば知ることができる。

M243 ・ M223 と同じ。

MZB —

BUB ・ 本文と同じ。ただし、値は6Kバイト単位。

FM8 ・ 本文と同じ。

PSP —

PC88 DSKF (1), 機能
・ 機能を指定することにより、さらに詳しい情報も返される。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT —

HC ・ ディスク・ナンバは“A : ”
(A は B, C, D でも可) である。

FP11 ・ 本文と同じ。

SMC ・ CPM と同じ。

MZ35 ・ PC3 と同じ。

X1Hu DEVF (“0 : ”)
・ 外部記憶デバイスの未使用領域のクラスタ数 (1 クラスタ=4096 バイト) を与える。

MB16 ・ DOS レベルで外部コマンドとして CHKDSK がある。

IBM55 ・ MB16 と同じ。

PC100 ・ PC88 と同じ。

MSX —

ファイルの終わりの検出 EOF

エンド・オブ・ファイル

形式：EOF (5)

機能：シーケンシャル・ファイルが終わりの時真となる。

- 特徴：
- 1 番号はファイル番号。
 - 2 シーケンシャル・ファイルを読む前に、EOF でファイルの終わりになったかどうかを調べ、エラーの発生を防ぐのに便利である。また、あらかじめレコード数のわかっていないファイルを読むとき使われる。
 - 3 最後のレコードを読んだとき真になる。機種によっては(C180など)、そのとき真にならず、さらにもう1レコード読もうとしたとき真になるので注意。

プログラム例：EOF を使ってファイル“AFILE”の内容をすべて出力する。

```
10 OPEN "AFILE" FOR INPUT AS #5
20 IF EOF(5) THEN GOTO 50
30 INPUT #5, A$: PRINT A$
40 GOTO 20
50 PRINT "End": CLOSE 5
60 END
```

実行例

```
Test file. 1'st line.
Test file. 2'nd line.
Test file. 3'rd line.
Test file. 4'th line.
Test file. 5'th line.
Test file. last line.
End
```

参照：AT EOF #n

APL ・該当するステートメントはない。ただし、ON ERR GOTO文で処理できる。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・APL と同じ。

MZK IF EOF (#1) THEN 100
・必ず IF 文として用いる。

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・M223 と同じ。

IF8² ・本文と同じ。

レベル₃ ・指定したファイルがコミュニケーション・ファイルである場合、バッファが空の時も真となる。

C180 AT EOF #1: 300 とすると、ファイルが終わりのとき行番号 300 へ飛ぶ。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・レベル₃ と同じ。

PSP GET # 15, A1, B1\$ EOF
1000

・15はファイル・コード、A1, B1\$ は変数 (二つ以上繰り返し、省略可)、EOF 行番号 (省略可)、
・行番号はファイルの終わりを検出

したときに分岐する所、指定がなければ NO-DATA のエラー。

PC88 ・レベル₃ と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・レベル₃ と同じ。

FP11 ・本文と同じ。

SMC ・本文と同じ。

MZ35 ・M223 と同じ。

X1Hu ・本文と同じ。

MB16 ・レベル₃ と同じ。

IBM55 ・レベル₃ と同じ。

PC100 ・レベル₃ と同じ。

MSX ・本文と同じ。

レコード終端コードの指定 EOR #n

エンド・オブ・レコード #n

形式：EOR #6:'53'

↑ —
必要 ↑ レコード終端コード (2進数値)
ファイル
番号

機能：可変長レコードのレコード終端コードを指定する。

- 特徴：1 この文を実行する前と、この文によってレコード終端コードを '100' 以上に指定した場合は、'0A' をレコード終端コードとする。
- 2 フロント・インサート付プリンタを使うとき、行の終わりを '0D' にするのに便利である (プリンタの終端コード)。

プログラム例：“CFILE”にデータ“1ABCD2ABCD3ABCD”を出力し、EOFを変えてデータを入力した例。'41'は“A”，'44'は“D”のアスキー・コードである。

```
100 OPEN #9:"CFILE.DT",INOUT,VARIABLE(20)
110 E$="1ABCD2ABCD3ABCD"
120 OUTPUT #9:E$
130 RESTORE #9
140 INPUT #9:E$
150 PRINT E$
160 RESTORE #9
170 EOR #9:'41'
180 INPUT #9:E$
190 PRINT E$
200 RESTORE #9
210 EOR #9:'44'
220 INPUT #9:E$
230 PRINT E$
240 CLOSE #9
250 END
```

実行例

```
1ABCD2ABCD3ABCD
1
1ABC
```

参照：

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 SEP #6, &53
・機能は本文と同じ。

IF8₂ —レベル₃ —

C180 ・本文参照。

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ・PC3と同じ。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

ファイルの属性表示 ATTR\$

アトリビュート\$

形式: ATTR\$ (M)

機能: 引数のドライブ、バッファ、ファイルの現在の属性を示す属性文字を返す。

特徴: 1 引数はドライブ番号、#ファイル番号、ファイル名とする。
2 属性と文字列の関係

書き込み禁止	"P"
書き込み確認	"R"
属性なし	" "

3 属性は SET 命令により設定する。

用語: ¹カタログ ボリューム内に納められているファイル名や使用容量、属性などを画面にリストとして出すこと。

プログラム例: "afile" に属性を設定し、それを表示する。

```
10 SET "afile", "R"
20 PRINT "attribute="; ATTR$("afile")
30 SET "afile", "A"
40 PRINT "attribute="; ATTR$("afile")
50 SET "afile", "P"
60 PRINT "attribute="; ATTR$("afile")
70 END
```

実行例

```
attribute=R
attribute=
attribute= P
```

参照: SET

APL ¹・カタログ・リスト表示の際、ファイル・モード記号の次に(*)マークが付くと、書き込み禁止状態である。

TRS₁ —

CPM ¹・本文と同じ。

M223 ¹・属性は、CLI コマンドで LIST 時に表示される。
・Wは書き込み禁止、Rは読み出し禁止、Pは属性変更禁止。

MZK ¹・ディレクトリ表示の際、ファイル・モード記号の次に(*)マークが付くと、書き込み禁止である。

CBM —

PC8 ¹・本文参照。

TRS₁₁ —

PC3 ¹・カタログ・リスト表示の際、ファイル名の次に(*)マークが付くと、書き込み禁止である。

IF8₂ ¹・本文と同じ。

レベル₃ —

C180 ¹・属性は、CLI コマンドの DIRECT で DE を指定すれば表示される。
・Rは読み出し禁止、Wは書き込み禁止、Dはファイルの削除禁止、Nはファイル名種別の変更禁止、Aはファイル属性の変更禁止。

M243 ¹・M223 と同じ。

MZB ¹・MZK と同じ。

BUB ¹・本文と同じ。

FM8 —

PSP —

PC88 ¹・本文と同じ。

N52 ¹・#ファイル番号を指定するとき、ファイルはオープンされていなければならないが、ファイル名を指定するときはファイルがオープンされていなくてもよい。
・Rの表示はない。

PC6 —

MLT —

HC —

FP11 —

SMC ¹・DIRを実行すると表示される。
・Rは書き込み禁止、Pはシステム上の DIR コマンド時の表示禁止。

MZ35 ¹・PC3 と同じ。
・これに関連して、ディスク・ユーザ領域を変更する USER 命令、操作対象ディスクを切り替える命令もある。

X1Hu ¹・引数は“ファイル・ディレクトリ: ファイル名”でファイルの属性を与える。属性はPが書き込み禁止、Rが書き込み確認。

MB16 —

IBM55 —

PC100 —

ファイルの属性表示 ATTR\$

アトリビュート\$

MSX —			

現在のセクタ番号の表示 FPOS

ファイル・ポジション

形式：FPOS (M)

機能：ディスク・ファイル中の番号Mのファイルが格納されている物理的なセクタ番号を与える関数。

- 特徴：1 Mはファイル番号。
2 セクタ番号はトラック0、ヘッド0、セクタ1を0番とし、順に1きざみで数える。
3 ディスケットの中をユーザが直接操作するときに便利である。

プログラム例：“afile”から2文字ずつ読み込み数値に変換して出力、そのときのセクタ番号とレコード番号を出力する。

```
10 OPEN "afile" AS #1
20 FIELD #1,2 AS E$
30 FOR I=1 TO 5
40 GET #1
50 A%=CVI(E$)
60 LPRINT "e$=";E$;"   cvi=";A%;
70 LPRINT "   fpos=";FPOS(1);
80 LPRINT "   loc=";LOC(1)
90 NEXT I
100 CLOSE #1
110 END
```

実行例

```
e$=ta   cvi= 24948   fpos= 81   loc= 1
e$=ke   cvi= 25963   fpos= 82   loc= 2
e$=ch   cvi= 26723   fpos= 83   loc= 3
e$=an   cvi= 28257   fpos= 84   loc= 4
e$=?!   cvi= 8511    fpos= 85   loc= 5
```

参照：LOC

APL —

TRS_I —

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ ・本文と同じ。

レベル₃ —

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 —

PSP —

PC88 ・ディスク・ファイルのみでなく、Mで指定されたファイルが最後に読み書きした位置を与える。

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu ・本文と同じ。

MB16 —

IBM55 —

PC100 —

MSX —

レコード番号の表示 LOC

ロケーション・カウンタ

形式: LOC (1)

機能: 現在のレコード番号を与える関数。

- 特徴:
- 1 引数はファイル番号。
 - 2 ランダム・ファイルの場合は、レコード番号を省略した GET または PUT 文が実行された時に使用される次のレコード番号が与えられる。
 - 3 シーケンシャル・ファイルの場合は、これまでに使用されたセクタ数が与えられる。
 - 4 読み出したレコード数を知るのに便利である。

プログラム例: "afile" から 2 文字ずつ読み込み数値に変換して出力し、そのときのセクタ番号とレコードを出力する (実行例は LOF 参照)。

```
10 OPEN "afile" AS #1
20 FIELD #1,2 AS E$
30 FOR I=1 TO 5
40 GET #1
50 A%=CVI(E$)
60 PRINT "e$=";E$;"   cvi=";A%;
70 PRINT "   fpos=";FPOS(1);
80 PRINT "   loc=";LOC(1)
90 NEXT I
100 CLOSE #1
110 END
```

参照: FPOS

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。ただし、ファイルがランダム・アクセスで OPEN されていないとエラーになる。

C180 RPOS (1)
・機能は本文と同じ。

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。ただし、シーケンシャル・ファイルには用いられない。

PSP —

PC88 ・本文と同じ。ただし、ファイル中での論理的な現在位置を与える。
・キーボード・ファイルの場合は、キーボード・バッファにある文字

数を返す。
・RS-232 C コミュニケーション・ファイルの場合は、入力バッファ中にある文字数を返す。

N52 LOC (#1)
・機能は本文と同じ。# は省略可。
・OPEN 直後のこの関数の値は 0 である。

PC6 ・本文と同じ。

MLT ・特徴 2 は、最後の GET、PUT の時のレコード番号が与えられる。また、特徴 3 は、ファイルの終端位置を 0 から始まるブロック位置番号で与える (1 ブロック = 128 バイト)。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC REC (1)
・シーケンシャル・ファイルには用いられない。

MZ35 —

X1Hu ・シーケンシャル・ファイルはブロック数を返す。

MB16 ・シーケンシャル・ファイルの入力モードで OPEN された場合、ファイルに入力する前でも 1 を返す。コミュニケーション・ファイルを指定した場合は、入力バッファにたまっている文字数。

IBM55 ・特徴 2 は、最後の GET、PUT のときのレコード番号。
・特徴 3 は、これまで使用されたレコード数を与える (1 レコード = 128 バイト)。

ファイルの大きさの表示 LOF

レングス・オブ・ファイル

形式: LOF (1)

機能: ファイルにおける最後のレコード番号, つまり最大のレコード番号を与える関数。

- 特徴: 1 引数はファイル番号。
 2 ランダム・ファイルの場合は, PUT または GET 文によりアクセスされた最大のレコード番号が与えられる。
 3 シーケンシャル・ファイルの場合は, そのファイルが使用しているセクタ数を与える。
 4 ファイルの大きさを知るのに用いる。

用語: ¹⁾ROM カートリッジ プログラムやデータを ROM 化し, カートリッジにしたもの, これを利用すると, RAM をほかの目的に使うことができる。

プログラム例: 整数型データを読み込み, 文字に変換して出力し, "afile" に格納する。そのときのレコード番号を出力し, 最大のレコード番号も出力する (プログラム例は LOC を参照)。

実行例

```
e$=   cui= 0
e$=ta   cui= 24948   fpos= 81   loc= 1
e$=ke   cui= 25963   fpos= 82   loc= 2
e$=ch   cui= 26723   fpos= 83   loc= 3
e$=an   cui= 28257   fpos= 84   loc= 4
e$=?!   cui= 8511    fpos= 85   loc= 5
```

参照: LOC

APL —

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ, ただし, ファイルがランダム・アクセスで OPEN されていないとエラーになる。

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・ランダム・ファイルについては本文と同じ。
 ・その他の時は, 入力バッファ中の文字数を与える関数となる。キーボードに割り当てられたファイル番号に対しては利用できない。

PSP —

PC88 ・本文と同じ, ただし, ディスク・ファイル以外でも可能。
 ・RS-232 C コミュニケーション・

ファイルの場合には, バッファの残りバイト数を返す。

N52 ・シーケンシャル・ファイルには用いられない。

PC6 ・ファイル番号で指定されたランダム・ファイルのレコード数を求める関数。

MLT —

HC ・ROM カートリッジ¹⁾の場合は残りの長さを, RS-232C ポートの場合はバッファ内のデータの数をバイト数で返す。

FP11 ・本文と同じ。

SMC —

MZ35 —

X1Hu ・シーケンシャル・ファイルはブロック数を返す。

MB16 ・コミュニケーション・ファイルでは, 入力バッファの残りバイト数, 入力バッファは 512 バイト。

IBM55 ・ファイルに割り振られたバイト数を返す。

PC100 ・最大の大きさをバイト数で返す。RS-232C コミュニケーション・ファイルの場合には, バッファの残りバイト数を返す。

MSX —

カセット・ロード CLOAD

形式：CLOAD “test”

プログラム名

機能：カセット・テープからプログラムを読み込む。

- 特徴：
- 1 引用符(“”)で囲まれたプログラム名を検索する。もし見つけた時は、“Found: プログラム名”と出力し、画面右上に星印を点滅させる。名前が違うプログラムを見つけた時は、そのプログラム名を、“Skip: プログラム名”と出力する。
 - 2 プログラム名は省略できない。
 - 3 この命令を実行すると、初めにメモリを内容クリアする(プログラムは破壊される)。
 - 4 (STOP)キーを押すとこの動作は中止され“Tape read Error”と表示される。
 - 5 カセット・テープに格納したプログラムを読み出すのに使う。

実行例：TEST という名前のプログラムをテープからロードする。

```
cload"TEST"
Found:TEST
Ok
```

参照：CLOAD ?, CSAVE

カセット・ロード

APL LOAD

- ・プログラム名は指定しない。再生の初めと終わりにブザー音を出す。

TRS_I ・本文と同じ。

CPM ・本文と同じ。ただし、CLOAD*で配列も読み込める。

M223 —

MZK LOAD / T “TEST”

- ・機能は本文と同じ。プログラム名は省略可。

CBM LOAD “TEST”, 1

- ・1は省略できる。プログラム名を指定しないと、テープ上の最初のプログラムを読み込む。

PC8 ・本文参照。

TRS_{II} —

PC3 CLOAD TEST

- ・プログラム名に引用符(“)はいらない。
- ・データ、プログラムの双方ともロードできる。
- ・プログラムのロードのとき、データはクリアされない。
- ・プログラム名を省略すると、最初に見つけたプログラムをロードする。
- ・データのロードのとき、後に変数名を付ける。このときプログラムはクリアされない。
- ・データ名を省略すると、最初に見つけたデータを変数にロードする。

IF8₂ LOAD “CAS1: AFILE”

- ・本文と同じ。CAS1: は省略できる。

レベル₃ LOAD “CAS0: TEST”

- ・CAS0: は省略できる。
- ・プログラム名を省略すると、最初に見つけたプログラムをロードする。、R を付けるとロードした後すぐ実行する。
- ・LO₂ という省略型も使える。

C180 —

M243 —

MZB ・MZK と同じ。

BUB ・本文と同じ。テープ・レコーダを再生状態にしてから実行する。

FM₈ LOAD “CAS0: TEST”

- ・レベル₃と同じ。ただし、ROM モードのときのみ CAS0: は省略できる。また、プログラム名は省略できない。

PSP ・本文と同じ。ただし、ロード開始アドレスは、そのプログラムの CSAVE 時の開始アドレスと一致している。また、@ で現在の BASIC 使用限度領域に合わせてロードできる。

PC88 LOAD “CAS2: TEST”

- ・本文と同じ。ただし、プログラム名の前に CAS1: を付けると1200 ボー、CAS2: で 600 ボーの速さでテープからロードする。初めにすべての開いているファイルを閉じるが、R を付けると開いたまままでロード後すぐ実行する。また、指定されたプログラム名を見つけたまでは、メモリ内容を保存する。

N52 —

PC6 ・プログラム名を省略すると最初のプログラムを読み込む。

カセット・ロード CLOAD

カセット・ロード

<p>・CLOAD*配列名 とするとデータの読み込みも可能。</p>	<p>MSX ・プログラム名を省略すると最初のプログラムを読み込む。</p>	
<p>MLT —</p>		
<p>HC LOAD "CAS1: PROG1, ASC"</p> <ul style="list-style-type: none"> ・CAS1 はデバイス名。 ・PROG1 はプログラム名。 ・特徴1 以外は本文と同じ。 		
<p>FP11 LOAD "CAS0: TEST"</p> <ul style="list-style-type: none"> ・フロッピー・ディスクが接続されていない時CAS0: は省略できる。 ・プログラム名の前に (F) を付けると1200ボー, (S) を付けると300 ボーの速さでテープからロードする。省略すると本体の仕様設定スイッチの指定に従う。 ・プログラム名を省略すると最初に見つけたプログラムをロードする。 		
<p>SMC CLOAD "CTR: TEST, BAS"</p> <ul style="list-style-type: none"> ・デバイス名を指定することによりカセット・テープ以外にも使用できる。 ・省略すると CRT が指定される。ただし、プログラムが見つからなければメモリ内容は保持される。 ・SAVE 時に指定すれば AUTO スタートができる。 ・LOAD も使用できる。 		
<p>MZ35 —</p>		
<p>X1Hu LOAD "CAS: TEST"</p> <ul style="list-style-type: none"> ・ファイル名を省略すると、最初に見つけたファイルをロードする。 		
<p>MB16 —</p>		
<p>IBM55 —</p>		
<p>PC100 —</p>		

カセット・セーブ CSAVE

形式：CSAVE "test"

機能：メモリ上の BASIC プログラムをカセット・テープに格納する。

- 特徴：1 引用符（"）で囲まれたプログラム名で、プログラムを格納する。
2 プログラム名は何文字でもよいが、7文字目以降は無視される。

実行例：TEST という名前で BASIC のプログラムをカセットテープに格納する。

```
list
10 REM test program
20 PRINT"MY PROGRAM NAME IS TEST"
30 END
Ok

csave"TEST"
Ok
```

参照：CLOAD, CLOAD?

カセット・セーブ

APL SAVE

- ・本文と同じ。ただし、プログラム名は指定しない。またセーブの始めと終わりにブザー音を出す。

TRS1 ・本文と同じ。

CPM ・本文と同じ。ただし、プログラム名として有効なのは、最初の一文字のみであとは無視される。

M223 —

MZK SAVE/T "TEST"

- ・機能は本文と同じ。

CBM SAVE "TEST", 1, 0

- ・本文と同じ。ただし、第1の引数、1は省略可。また第2の引数は1のときセーブの終わりにEOT(エンド・オブ・テープ)が書き込まれ、0のとき書かれない。省略すると0に設定される。

PC8 ・本文参照。

TRSII —

PC3 CSAVE TEST

- ・記録開始時、完了時に信号音を出す。
- ・変数データをプログラム名を付けてセーブできる。
- ・配列変数は A (*) という型でセーブする。
- ・機械語をセーブするには、プログラム名を指定して先頭アドレス、バイト数を指定する。
- ・CSAVE@ とするとリスト出力、修正不可能な形でセーブできる。
- ・プログラム名は省略できない。ほかは本文と同じ。

IF82 SAVE "CAS1: TEST"

- ・CAS1: は省略できる。ほかは本文と同じ。

レベル3 SAVE "CAS0: TEST", A

- ・CAS0: は省略可。
- ・後に、A を付けるとアスキー型式でプログラムをセーブする。また、B を付けるか省略すると普通の形(内部表現型式)でプログラムをセーブする。
- ・省略型 SA、も使える。

C180 —

M243 —

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・レベル3 と同じ。

PSP CSAVE TEST

- ・機能は本文と同じ。ただし、プログラム名は最大8文字有効である。また引用符(")でプログラム名を囲う必要はない。

PC88 SAVE "CAS2: TEST"

- ・プログラム名の前に CAS1: とすると1200ボー、CAS2: とすると600ボーで格納される。
- ・後に、A を付けるとアスキー型式で格納され、P を付けるとPC3のCSAVE@ と同じ。

N52 —

PC6 ・本文と同じ。

- ・CSAVE*配列名 とすると、データのセーブも可能。

MLT —

カセット・セーブ CSAVE

カセット・セーブ

<p>HC SAVE "CAS0 : ABC"</p> <ul style="list-style-type: none"> ・文末に A を付けるとアスキー型式でセーブされる。 	<ul style="list-style-type: none"> ・ M1 はボーレートで、1 のとき 1200 ボー、2 のとき 2400 ボー。省略したとき、SCREEN 文のボーレート設定値となる。 ・アスキー型式でセーブするときは SAVE 文を使う。 		
<p>FP11 SAVE "CAS0 : TEST", A</p> <ul style="list-style-type: none"> ・文末に A を付けるとアスキー型式でプログラムをセーブする。省略すると内部型式。 			
<p>SMC CSAVE "CTR : TEST. BAS", 先頭アドレス, 最終アドレス</p> <ul style="list-style-type: none"> ・引用符の前に /A を付けると、アスキー型式で格納され、/P を付けると PC3 の CSAVE@ と同じことができる。/C を付けると任意のコマンドも同時セーブできる。 ・デバイス名を指定することにより、カセット以外にも使用できる。デバイス名を省略するとカセットが指定される。 ・タイプ名を省略すると BAS が指定される。 ・先頭アドレスと最終アドレスで指定されたアドレス間のメモリの内容が BASIC プログラムと同時に格納される。 			
<p>MZ35 —</p>			
<p>XIHu SAVE "CAS : TEST", A</p> <ul style="list-style-type: none"> ・オプションで A を指定すると、アスキー型式となり、省略するとバイナリ型式となる。 			
<p>MB16 —</p>			
<p>IBM55 —</p>			
<p>PC100 —</p>			
<p>MSX CSAVE "ファイル名", M1</p>			

カセット・ロード・ベリファイ¹⁾ CLOAD?

カセット・ロード?

<p>形式：CLOAD? "test"</p> <p style="text-align: center;">└───┘ プログラム名</p> <p>機能：メモリ上のプログラムと、セーブしたカセット・テープ上のプログラムとの比較検証（うまく格納されているかのテスト）を行う。</p> <p>特徴：1 プログラム名は6文字まで有効で、それ以降は無視される。 2 メモリ内のプログラムと、引用符(")で囲まれた名前をもつカセット・テープ上のプログラムとを比較して、等しければ"Ok"、違っていれば"Bad"と画面に出力される。 3 カセット・テープ上のプログラムはロードされない。 4 プログラムが正しくテープに格納されたかを確認するのに便利である。</p> <p>用語：¹⁾ベリファイ 一つのデータをコピーしたとき、もとのデータとコピーしたデータとを比較し確かめること。 ²⁾チェック・サム あるデータのまとまりにおいて、それぞれ16進を加算すること、誤りの検出に用いる。</p>	<p>APL —</p> <p>TRS_I ・本文と同じ。ただし、ファイル名を指定しないと、テープ上の最初のプログラムと比較する。</p> <p>CPM ・本文と同じ。</p> <p>M223 —</p> <p>MZK VERIFY "TEST" ・TRS_Iと同じ。</p> <p>CBM VERIFY "TEST", 1 ・本文と同じ。ただし、1は省略でき、またプログラム名を省略すると(00)₁₆(ナル・コード)が設定される。</p> <p>PC8 ・本文参照。</p> <p>TRS_{II} —</p> <p>PC₃ CVERIFY TEST ・使用法は CLOAD の場合と同様。</p> <p>IF8₂ ・本文と同じ。</p>	<p>PSP ・一致しない時エラー・メッセージを出力し、処理を中断する。</p> <p>PC88 LOAD? "CAS2:TEST" ・本文と同じ。ただし、プログラム名の前にCAS1:, CAS2:を付ける。すると、1200/600ボーで比較する。</p> <p>N52 —</p> <p>PC6 ・TRS_Iと同じ。</p> <p>MLT —</p> <p>HC LOAD "CAS1:PROG1,ASC" ・実行中一致しない場合 IO Error が出る。</p> <p>FP11 VERIFY "CAS0:(F)STAR" ・ファイル自身に含まれるパリティとチェック・サム²⁾をもとにカセット・テープ・レコーダ上のファイルをチェックする。</p>
<p>実行例：TESTという名前のテープ上のプログラムとメモリ上のプログラムを照合する。</p> <pre> cload?"test" Skip:TEST Tape read ERROR Ok </pre> <p style="text-align: right;">*</p> <pre> cload?"TEST" Found:TEST Ok </pre> <p>参照：CLOAD, CSAVE</p>	<p>レベル₃ LOAD? "CAS0:TEST" ・TRS_Iと同じ。ただし、CAS0:は省略できる。</p> <p>C180 —</p> <p>M243 —</p> <p>MZB ・MZKと同じ。</p> <p>BUB ・テープ・レコーダを再生状態にしてから実行する。</p> <p>FM8 ・レベル₃と同じ。ただし、"CAS0:..."は省略できる。プログラム名は8文字まで。</p>	<p>SMC CLOAD/V "CTR:TEST" ・本文と異なる点はCLOADを参照。 ・LOAD/V "CTR:TEST"も使用できる。</p> <p>MZ35 —</p> <p>X1Hu LOAD? "CAS:TEST" ・メモリ上のプログラムと指定されたファイルとの照合を行う。 ・引数を省略すると、最初に見つけたファイルと照合する。 ・VERIFYと同じこと。</p> <p>MB16 —</p> <p>IBM55 —</p>

カセット・ロード・ベリファイ CLOAD?

カセット・ロード?

PC100 —			
MSX ・TRS ₁ と同じ。			

カセット・テープのリード・オープン ROPEN

リード・オープン

形式：ROPEN “FILE”

省略可

ファイル名

機能：カセット・テープのデータの読み出しをするため、ファイルをオープンする。

- 特徴：**
- 1 データ専用のファイルとしてオープンする。
 - 2 プログラムの読み出しには LOAD がある。
 - 3 ROPEN を実行したあとは必ず CLOSE しなければならない。
 - 4 カセット・テープ中からデータを読み出すとき、INPUT/T 命令の前にこの命令をおく。

プログラム例：“TEXT” というファイルを読むためにカセット・テープをオープンする。

```
10 ROPEN "TEXT"
20 INPUT/T X$
30 PRINT X$
40 CLOSE
50 END
```

実行例

```
PLAY
FOUND TEXT
LOADING TEXT
ABCDEF
```

参照：OPEN, LOAD, WOPEN, CLOSE

APL OPEN

・機能は本文と同じ。

TRS₁ —

CPM —

M223 —

MZK ・本文参照。

CBM OPEN n, 1, “ファイル名, S”

・ただし、(n) はロジカル・ファイル番号。1 はデバイス番号でカセットのときは 1 または 2。またファイル名の後の S を指定しないとプログラム・ファイルになる。シーケンシャル・ファイルのとき W を指定しないと READ とみなされる。

PC8 —

TRS₁₁ —

PC3 —

IF8₂ ・PC88 と同じ。

レベル₃ OPEN “I”, ファイル番号, “CAS 0 :”
・機能は本文と同じ。

C180 —

M243 —

MZB ROPEN/T
・機能は本文と同じ。

BUB —

FM8 OPEN “I”, ファイル番号, “CAS0 : ファイル名”
・機能は本文と同じ。

PSP —

PC88 OPEN “CAS1 : FILE”
FOR INPUT AS 1
・機能は本文と同じ。

N52 —

PC6 —

MLT —

HC ・レベル₃ と同じ。

FP11 OPEN “CAS0 : FILE”
FOR INPUT AS #1
・機能は本文と同じ。

SMC OPEN/I “CTR : AFILE”
OPEN/R “CTR : BFILE”
・機能は本文と同じ。

MZ35 —

X1Hu OPEN “I”, # 1, “CAS : TEST”
・機能は本文と同じ。

MB16 —

IBM55 —

PC100 —

MSX OPEN “CAS : ファイル名”
FOR INPUT AS # ファイル番号
・機能は本文と同じ。

カセット・テープのライト・オープン WOPEN

ライト・オープン

形式：WOPEN "FILE"

省略可

ファイル名

機能：カセット・テープに、数値変数やストリング¹⁾変数の内容を書き込むため、ファイルをオープンする。

特徴：1 データ専用のファイルとしてオープンする。
 2 プログラムの記憶には SAVE がある。
 3 WOPEN のあとはかならず CLOSE しなければならない。
 4 カセット・テープ中へデータを書き込むとき、PRINT/T 命令の前にこの命令をおく。

用語：¹⁾ストリング 文字列のこと。（"）でくくることによりほかの情報と区別する。

プログラム例：文字列"ABCDEF"を"TEXT"というファイルに書き込むためにカセット・テープをオープンする。

```
10 WOPEN "TEXT"
20 X$="ABCDEF"
30 PRINT/T X$
40 CLOSE
```

実行例

```
RECORD.PLAY
WRITING TEXT
READY
```

参照：OPEN, SAVE, ROPEN, CLOSE

APL OPEN
 ・機能は本文と同じ。

TRS_I —

CPM —

M223 —

MZK ・本文参照。

CBM OPEN n, 1, "ファイル名,
 S, W"

・ただし、n はロジカル・ファイル番号、1 がデバイス番号でカセットのときは1または2。また、ファイル名の後のSを指定しないとプログラム・ファイルになる。シーケンシャル・ファイルのときWを指定しないと READ とみなされる。

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・PC88と同じ。

レベル₃ OPEN "O", ファイル番号, "CAS 0 :"
 ・機能は本文と同じ。

C180 —

M243 —

MZB WOPEN/T
 ・機能は本文と同じ。
 ・この命令はディスクでも使う。

BUB —

FM8 OPEN "O", ファイル番号,

"CAS0: ファイル名"
 ・機能は本文と同じ。

PSP —

PC88 OPEN "CAS1: FILE"
 FOR OUTPUT AS 1
 ・機能は本文と同じ。

N52 —

PC6 —

MLT —

HC ・レベル₃と同じ。

FP11 OPEN "CAS0: FILE"
 FOR OUTPUT AS #1
 ・機能は本文と同じ。

SMC OPEN/O "CTR: AFILE"
 OPEN/S "CTR: AFILE"
 ・機能は本文と同じ。

MZ35 —

X1H_u OPEN "O", #1, "CAS :
 TEST"
 ・機能は本文と同じ。

MB16 —

IBM55 —

PC100 —

MSX OPEN "CAS: ファイル名"
 FOR OUTPUT AS # ファイル
 番号
 ・機能は本文と同じ。

カセット・テープ・レコーダのモータ制御 MOTOR

モータ

<p>形式：MOTOR1 □ 省略可</p> <p>機能：カセット・テープ・レコーダ (CT) のモータの制御。</p> <p>特徴：1 MOTOR0 でモータを OFF, MOTOR1 でモータを ON する。 2 引数を省略すると、状態を反転する。すなわちモータが ON のとき OFF, OFF のとき ON にする。 3 CSAVE, CLOAD 実行の前に、MOTOR0 と使うと便利である。</p>	<p>APL —</p> <p>TRS₁ ・ OUT 255, 0 でオン, OUT 255, 4 でオフ。</p> <p>CPM —</p> <p>M223 —</p> <p>MZK —</p> <p>CBM —</p> <p>PC8 ・ 本文参照。</p> <p>TRS₁₁ —</p> <p>PC3 —</p> <p>IF8₂ ・ 本文と同じ。</p> <p>レベル₃ ・ MOTOR ON でモータを オン, MOTOR OFF でモータを オフにする。</p> <p>C180 —</p> <p>M243 —</p> <p>MZB ・ 該当なし。ただし, REW, FAST でカセット・テープの早送 り / 巻きもどしができる。</p> <p>BUB ・ 本文と同じ。</p> <p>FM8 ・ レベル₃ と同じ。 ・ SKIPF によりカセット・テープ の頭出しが可。</p> <p>PSP ・ MOTOR0 でモータを ON, 引数にそれ以外の数値を使うと, モータは OFF になる。引数は省 略できない。</p> <p>PC88 ・ 本文と同じ。</p>	<p>N52 —</p> <p>PC6 —</p> <p>MLT —</p> <p>HC ・ レベル₃ と同じ。 ・ これに関連して、マイクロ・カセ ットを制御する WIND, TAPCNT 命令がある。</p> <p>FP11 ・ レベル₃ と同じ。</p> <p>SMC ・ レベル₃ と同じ。</p> <p>MZ35 —</p> <p>X1Hu CSTOP ・ テープ走行をストップさせる。 CMT = M ・ カセットの状態を設定する。 M = 1 はリモートを OFF にする。 M = 2 はリモートを ON にする。</p> <p>MB16 —</p> <p>IBM55 —</p> <p>PC100 —</p> <p>MSX ・ 本文と同じ。</p>
<p>実行例：motor 0 で CT のモータを OFF にした後で、"TEST" を CT からロードする。</p> <pre> motor 0 Ok cload "TEST" Found: TEST Ok </pre> <p>※</p> <p>参照：CSAVE, CLOAD, CLOAD?</p>		

画面行制御 CONSOLE

コンソール

形式: CONSOLE 0, 25, 0, 1

0	25	0	1
---	----	---	---

省略可

機能: 画面の表示に対する機能を指定する。

- 特徴: 1 第1の引数でスクロール¹⁾の開始行を指定し、第2の引数でスクロールの行数を指定する。
- 2 第3の引数はファンクション・キー表示スイッチで、1で画面の最下段のファンクション・キーの内容を表示し、0で表示しない。
- 3 第4の引数はカラー/白黒スイッチで、1で画面をカラー・モード0で白黒モードにする。

用語: ¹⁾スクロール 画面の最後の行にカーソルが移動した時改行を行うと、最上位の行が消え、最下位の行が1行あがること。

プログラム例: 白黒モードのリバースで、メッセージを出力し、ノーマル画面にもどす。

```
10 WIDTH 80,20
20 CONSOLE 0,20,0,0:COLOR 4
30 PRINT "*** TEST PROGRAM
40 PRINT "    FOR CONSOLE COMMAND ***"
50 COLOR 0:END
```

実行例

```
*** TEST PROGRAM
    FOR CONSOLE COMMAND ***
```

参照: COLOR, KEY, WIDTH

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ ・本文と同じ。なお、第4の引数でスクロールの速さを調節できる(255で最もおそい)。省略すると0のジャンプ・スクロール(最も速い)に設定される。

レベル₃ ・本文と同じ。ただし、カラー/白黒表示スイッチはない。

C180 —

M243 —

MZB CONSOLE S0, 25
・最初の指定をSとすると、スクロールの開始行と終了行を示す。Cとするとディスプレイの行数を指定する(80または40)。Rとするとディスプレイをリバース、Nとするとノーマル表示にする。

BUB ・レベル₃と同じ。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・第4のパラメータで1はキー・クリック音ON、0でOFF;

MLT KEY ON KEY OFF
・KEY ONにより、ファンクション・キーの内容を画面最下段に表示し、KEY OFFによって、表示しない。

HC —

FP11 —

SMC ・第1の引数で1行表示文字数を指定する。
・第2の引数でスクロール開始行を指定し、第3の引数でスクロールの行数を指定する。
・第4の引数はスクロール・モードを指定する。
・直接コマンド・モードでは、スクロール・エリアは常に0~25行に設定される。

MZ35 —

X1Hu ・第1、第2の引数で垂直方向の表示開始ラインと表示ライン数を、第3、第4の引数で水平方向の表示開始カラムと表示カラム数を指定する。

MB16 ・MLTと同じ。

IBM55 ・MLTと同じ。

PC100 ・本文と同じ。ただし、第3、第4番目の引数はもたない。
・ファンクション・キー・スイッチの代わりにkey on off機能をもつ。

MSX ・MLTと同じ。

画面の消去 ERASE

イレース

形式：ERASE

機能：CRT の画面を消去する。

特徴：1 CRT 全画面を一度に消去するときに使う。

プログラム例：画面を消去する。

```
10 A=10
20 PRINT "A=";A
30 ERASE
40 PRINT "CRT ヲ ショウキョ シタ。"
50 END
```

実行例

CRT ヲ ショウキョ シタ。

参照：

APL ・該当なし。ただし、
CALL-958 で同様のことができる。

TRS_I CLS

CPM ・該当なし。ただし、
PRINT CHR\$(12) で同様のこ
とができる。

M223 ・該当なし。ただし、
PRINT CLEARで同様のこ
とができる。

MZK ・該当なし。ただし、
PRINT "C" で同様のこ
とができる。

CBM ・該当なし。ただし、
PRINT CHR\$(147) で同様のこ
とができる。

PC8 ・CPM と同じ。

TRS_{II} ・TRS_I と同じ。

PC3 ・該当なし。ただし、
DISP CLEAR で同様のこ
とができる。

IF8₂ ・TRS_I と同じ。

レベル₃ ・TRS_I と同じ。

C180 ・本文参照。

M243 ・M223 と同じ。

MZB ・MZK と同じ。また、
PRINT CHR\$(6), GRAPH
C によっても同様のこ
とができる。

BUB ・CPM と同じ。

FM8 ・該当なし。ただし、CLS

0 または CLS で同様のこ
とができる。

PSP ・本文と同じ。

PC88 ・該当なし。ただし、CLS
1 でテキスト画面を消去可能。
CLS2 とするとグラフィック画面
の表示部分だけ消去ができる。
CLS3 でテキスト画面、グラフィ
ック画面の両方を消去する。

N52 ・該当なし。ただし、
(FNC) キー + (クリア) キー、ま
たは、PRINT CHR\$(12) で同
様のこ
とができる。

PC6 ・CPM と同じ。また、
CLS も使える。

MLT ・TRS_I と同じ。

HC CLS
・仮想スクリーンのみをクリアする。
・PRINT CHR\$(12) でも可。
・GCLS でグラフィック画面のみの
クリアもできる。

FP11 TRS_I と同じ。

SMC CCLEAR N1, N2
・キャラクタ画面上の N1 行から N2
行の文字を消去する。
・N1 を省略すると最上行を指定し、
N2 を省略すると最下行までを指定
したことになる。
GCLEAR N1, M2
・-32768 ≤ M1 ≤ 32767, 0 ≤ M2 ≤ 4
の範囲でなければならない。
・M2 で指定された論理演算を行い、
グラフィック画面を、M1 で指定さ
れた色と画面の色でうめる。
・引数を省略すると、それ以前の
GCOLOR 文で指定してあるバック
・グラウンド・カラー、または、

画面の消去 ERASE

イレース

<p>論理演算コードが指定される。 GCOLOR 文がなければ M1=0, M2=0 が指定される。</p> <p>WIPE</p> <ul style="list-style-type: none"> ・キャラクタ表示面, グラフィック表示面を消去して, 画面表示を初期化する。 			
<p>MZ35 ・PC3 と同じ。</p> <p>CLS</p> <ul style="list-style-type: none"> ・グラフィック画面の消去を行う。 			
<p>X1Hu CLS n</p> <ul style="list-style-type: none"> ・画面を消して背景色のための画面にする。 ・n=0 グラフィック 1, 2, 3 n=1 グラフィック 1 n=2 グラフィック 2 n=3 グラフィック 3 n=4 グラフィック 1, 2, 3 とテキスト 省略はテキストのみを消す。 ・テキストは CONSOLE 文で指定されているときはその範囲。 ・グラフィックは WINDOW が指定されているときは絶対座標の範囲。 			
<p>MB16 ・該当なし。ただし, CLS 0 でグラフィック, テキスト両画面, CLS1 でグラフィック, CLS2 でテキスト画面を消去。</p>			
<p>IBM55 ・CLS で同様のことができる。CLS n において WIDTH 文で設定されている画面の部分だけを消すことができる。</p>			
<p>PC100 ・該当なし。ただし, CLS のみはスクロール・ウィンド内。CLS<数式>は画面全体のクリア。</p>			
<p>MSX ・TRS1 と同じ。</p>			

画面の文字数を設定 WIDTH

ウィドゥス

形式: WIDTH A1, A2

--	--

省略可

機能: 画面に表示する文字の行数と桁数を設定する。

- 特徴: 1 画面の行数 (A2) は、20と25を指定でき、1行の桁数 (A1) は、80、72、40と36を指定できる。
- 2 システム起動時の行数は20行であり、1行の桁数は40である。
- 3 これを実行すると、画面はクリアされる。
- 4 画面を見やすくする時などに使うと便利である。
- 5 引数を省略した場合は以前の値を保つ。

プログラム例: プログラム例のリストは一画面が20行、80文字/桁の場合である。実行例は一画面が行36文字/桁のものになっている。

```
10 WIDTH 36,25
20 FOR I=1 TO 5
30 PRINT STRING$(I,"?");I
40 NEXT I
50 END
```

実行例

```

  1
?? 2
??? 3
???? 4
????? 5
```

参照: MARGIN, MARGIN #n, POKE, CONSOLE

APL POKE 33, A1

POKE 35, A2

- ・A1で1行の桁数を指定、 $1 \leq A1 \leq 40$ 。
- ・A2で1画面の行数を指定、 $0 \leq A2 \leq 24$ 。

TRS: PRINT CHR\$(23)

- ・1行を32文字にできる。ただし、初期設定は64文字、行数は設定できない。

CPM WIDTH M

- ・ $15 \leq M \leq 255$ で、Mはキャリジ・リターンを出すまでである。
- ・初期設定は、1行72文字である。
- ・行数は設定できない。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS: PRINT CHR\$(M)

- ・Mは、1Eで1行80文字、1Fで1行40文字を設定する。
- ・行数は設定できない。

PC3 ・該当なし。ただし、CRT表示は1行40文字である。

IF8₂ ・本文と同じ。ただし、1行の桁数 (A1) は、80または40文字である。レベル₃ WIDTH M

- ・Mは、1行あたりの文字数で、40または80文字である。
- ・行数は設定できない。

C180 MARGIN A1+3

- ・算術式の値は、1~80。

- ・MERGINを実行しないと80となる。

M243 —

MZB CONSOLE C80

- ・1行80文字と1行40文字があり、1行40文字のときは、C40とする。
- ・行数は設定できない。

BUB ・本文と同じ。ただし、1行の桁数 (A1) は、80または40文字である。

・省略時は、1行80文字で25行である。

FM8 ・本文と同じ。ただし、1行の桁数 (A1) は、80または40文字である。

PSP ・本文と同じ。ただし、1行の桁数 (A1) と行数 (A2) の関係は、

A1	A2
80	25または20
36	24または19
40	6または8

- ・初期設定は36文字24行である。

PC88 ・本文と同じ。ただし、本文の機能以外に次のものがある。

① A1にファイル・ディスクリプタ、A2にサイズを与えると、A1に指定されたデバイス（プリンタまたはコミュニケーション・ポート）に対して、バッファのサイズを設定する。 $0 \leq A2 \leq 255$ 。初期設定は255である。

② A1に#ファイル番号、A2にサイズを与えると、A1で割り当てられているバッファに対して、その大きさをA2で指定できる。

画面の文字数を設定 WIDTH

ウィドウス

0 ≤ A2 ≤ 255. 初期設定は255.	
N52	—
PC6	—
MLT	・IF8 ₂ と同じ。ただし、A1とA2を省略した場合、実行しない。
HC	WIDTH A1, A2, A3 ・仮想スクリーンの大きさを、1行A1文字でA2行指定、A3はスクロール・マージンで、表示画面の左右の余幅。 ・A1は20～255、A2は4～255、A3は1～10の範囲とする。 ・外部ディスプレイにも命令は使える。
FP11	・レベル ₃ と同じ。 ・LWIDTH Mでプリンタの1行の長さを指定できる。文字数は80または132。
SMC	・CONSOLE文で指定できる。ただし、行数は25行だけである。
MZ35	CHANGE DISP' 文字定数を使う。
X1Hu	WIDTH 80 ・1行80字と1行40字がある。
MB16	・レベル ₃ と同じ。
IBM55	・1 ≤ A1 ≤ 80, 1 ≤ A2 ≤ 24 ・その他に、PC88の①、②の機能があるが、②にてファイル番号はデバイスでのみ指定できる。
PC100	・PC88と同じ。桁数、行数の組み合わせは次のとおり（桁×行）。

横表示	縦表示
80×25	64×45
80×32	85×45
80×64	85×90
90×25	PC100のディスプレイは縦にも横にも置くことができる。
90×32	
120×25	
120×32	
120×64	

・同様の命令にWIDTH USINGがある。
MSX WIDTH M1 ・M1の桁数を指定する。 ・40×24テキスト・モードで1～40。 ・32×24テキスト・モードで1～32。

スクローリングの指示 ROLL

ロール

形式：① ROLL ON

↑
必要

② ROLL OFF

↑

機能：CRT のスクローリングを指示する。

- 特徴：1 ROLL OFF のとき CRT の最下行にカーソルがある場合、改行すると 1 行目にカーソルを表示し、ROLL ON のとき CRT を 1 行スクロールしてカーソルを最下行に表示する。
- 2 カーソルが CRT の途中のとき、ROLL ON、ROLL OFF にかかわらず、改行により 1 行ずつカーソルが下がる。
- 3 ROLL 文が実行されなければ ROLL ON が暗黙に実行される。
- 4 SETFORM 文を実行すると、ROLL OFF となる。
- 5 CRT を固定するとき、ROLL OFF を使うと便利である。

プログラム例：スクローリングを指示する。

```
10 ROLL ON
20 FOR I=1 TO 29
30 PRINT I
40 NEXT I
50 END
```

実行例

```
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
```

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 ・ CONSOLE 0, 24
で同じこと可能。ただし、ROLL
OFF 機能はない。スクロールす
る範囲を指定できる。

TRS₁₁ —

PC3 —

IF8₂ ・ PC8 と同じ。レベル₃ ・ PC8 と同じ。

C180 ・ 本文参照。

M243 —

MZB ・ CONSOLE S0, 24
で同じこと可能。S の後の値を変
化することでスクロールする範囲
を指定可能。ただし、ROLL OFF
機能はない。

BUB ・ PC8 と同じ。

FM8 ・ PC8 と同じ。

PSP —

PC88 ・ PC8 と同じ。また、グラ
フィック画面のスクロールに
ROLL 命令を用いる。

N52 ・ PC8 と同じ。

PC6 ・ PC8 と同じ。

MLT —

HC SCROLL A1, A2, A3,
A4
・ A1 はスクロール・スピードで 0 ~
9, A2 は画面の移動の方法で 0 か
1, A3 はスクロール・ステップ X
で、LCD では 1 ~ 20, CRT では
1 ~ 32, A4 はスクロール・ステッ
プ Y で、LCD では 1 ~ 4, CRT で
は 1 ~ 16。
・ ROLL OFF の機能はない。

FP11 —

SMC CONSOLE ... 1
CONSOLE ... 0
・ 1 で ROLL ON, 0 で ROLL
OFF。

MZ35 SCROLL 0, 5, 5, 69, 14
・ テキスト座標 (5, 5) および (69, 14)
を対角点とする長方形内を上方向
へ 1 列分スクロールする。
・ スクロール方向は第 1 引数で与え
る。

X1Hu SCROLL 1
・ 引数が 0 または省略のとき、スク
ロールはストップする。
・ 引数が正の数のとき上方向、負の
数のとき下方向に、また値の絶対
値が大きいと 3 段階に速くなる。
ただし、TV 画面と混在のときの
み。

MB16 —

IBM55 —

PC100 ・ PC8 と同じ。

参照：SETFORM, CONSOLE

スクローリングの指示 ROLL

□ール

MSX —			

表示文字の読み込み SCREEN_{関数}

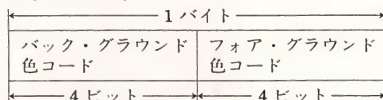
スクリーン

形式: SCREEN (M1, M2, A)

省略可

機能: 画面上に表示中の指定位置の文字コード, または色属性を与える.

- 特徴: 1 M1 で行位置 (1 から 25) を与え, M2 でカラム位置 (1 から 40 または 1 から 80) を与える.
- 2 A で文字コードまたは色属性のどちらを与えるのかを判断する. A を省略したとき, または 0 のときは現在の画面の色属性が次のようになる.



- 3 指定位置に文字が表示されなければ, 32 (空白) を示す.

プログラム例: 画面に文字列を表示し, 第 1 行第 1 列の文字コードを 16 進で求める.

```
LIST
10 FOR I=&H20 TO &HF7 STEP 20
20 FOR J=0 TO 19
30 PRINT CHR$(I+J);
40 NEXT J
50 PRINT
60 NEXT I
70 END
Ok
PRINT HEX$(SCREEN(1,1,0))
```

実行例

```
4C
Ok
```

参照:

APL ・該当なし. ただし, SCRN (M1, M2) はその場所のカラー・コードを与える.

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・A の指定はない.

レベル₃ SCREEN (M2, M1, A)
・A が 0 のときキャラクタ・モード.

C180 —

M243 —

MZB CHARACTER\$ (X, Y)
・カーソル位置 X, Y の文字を与える.

BUB —

FM8 ・本文と同じ. ただし, 属性のフォーマットは異なる.

PSP —

PC88 —

N52 —

PC6 SCREEN (M2, M1)

・SMC の CPOINT と同じ.

MLT ・本文参照.

HC —

FP11 —

SMC ・APOINT (M2, M1) で色属性を, CPOINT (M2, M1) で文字コードを与える.
・M1 は 0 から 24 まで, M2 は 0 から 39 または 0 から 79 の範囲.

MZ35 ・MZB と同じ.

X1Hu CHARACTER\$ (X, Y)
・カーソルの座標 X 列 Y 行の文字を与える.
SCRN\$ (X, Y, N)
・カーソルの座標 X 列 Y 行から N 文字の文字列を与える.

MB16 ・A が 0 あるいは省略のときはキャラクタ・コードを与える.

IBM55 ・行は 1 から 24, 桁は 1 から 80 の範囲.
・A が 0, または省略したときキャラクタ・モード.

PC100 ・A の指定はなく, 連続コードのみが返される.

MSX —

カーソル位置の設定 LOCATE

□ケート

形式：LOCATE 10, 10, 0
 └─┬─┬─┐
 水平 垂直 省略可
 位置 位置 位置
 カーソル・
 スイッチ

機能：CRT 画面上の任意の場所へカーソルを移動する。

特徴：1 通常カーソルは点滅しているが、カーソル・スイッチを 0 に指定することでカーソルを消すことができる。
 2 画面左上が 0, 0 である。
 3 垂直位置は 0 から 24 の範囲。
 4 水平位置は 0 から 79 の範囲。

プログラム例：画面上 (5, 0) の位置に (?), (5, 5) の位置に (*) を出力する。

```
10 LOCATE 5,0
20 PRINT "?"
30 LOCATE 5,5
40 PRINT "*"
50 END
```

実行例
run ?

*
Ok

参照：POS, CSRLIN

APL ・該当なし。ただし、VTAB, HTAB で代用可。この場合垂直位置の上限は 24、水平位置の上限は 255。

TRS_I —

CPM —

M223 CURSOR (10, 10)
 ・垂直位置の上限は 23、水平位置の上限は 79。
 ・PRINT 文中でのみ使用可。

MZK CURSOR 10, 10
 ・垂直位置の上限は 24、水平位置の上限は 39。

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 CURSOR 10, 10
 ・垂直位置の上限は 15、水平位置の上限は 39。

IF8₂ ・本文と同じ。ただし、水平位置の上限は、画面水平表示文字数 - 1、垂直位置の上限は 24。

レベル₃ ・省略形は LOC。
 ・垂直位置の上限は 24、水平位置の上限は 39 または 79。
 ・カーソル・スイッチは四つのモードがある (0 はノンプリンキング, 1 はカーソルなし, 2 は倍速プリンキング, 3 はノーマル・プリンキング)。

C180 CSR (10, 10)
 ・垂直位置は 1 から 24、水平位置は 1 から 80。
 ・PRINT 文中でのみ使用可。

M243 ・M223 と同じ。

MZB ・MZK と同じ。ただし、水平位置の上限は 39 または 79。

BUB ・本文と同じ。ただし、カーソル・スイッチを 1 に指定することでカーソルを消す。

FM8 ・本文と同じ。ただし、省略形は LOC。

PSP —

PC88 ・本文と同じ。ただし、座標の引数をそれぞれ省略できる。省略した場合、水平座標は 0、垂直座標は現在カーソルがセットされている行となる。

N52 ・本文と同じ。

PC6 ・垂直位置の上限は 15、水平位置の上限は 31。また、カーソル・スイッチはない。

MLT ・画面左上が 1, 1 である。また、垂直位置は 1 ~ 25、水平位置は 1 ~ 80。
 ・通常カーソルは点滅していない。カーソル・スイッチを 0 に指定するとカーソルが消える。

HC ・WIDTH 命令による仮想スクリーンをはみ出せない。また、垂直・水平位置とも 0 ~ 255 の範囲。

FP11 ・カーソル・スイッチはない。また、水平位置の上限は 39 または 79。
 ・CSR (10, 10) も使える。PRINT 文中で使われるほかは、本文と同じ機能。

SMC LOCATE (M1, M2)

カーソル位置の設定 LOCATE

ロケート

<ul style="list-style-type: none"> ・ M1 は水平位置で、0 から79の範囲。 ・ M2 は垂直位置で、0 から24の範囲。 <p>CURSOR ON</p> <ul style="list-style-type: none"> ・ カーソルが表示され、CURSOR OFF で消える。 ・ 直接コマンド・モード、コマンド待ち状態、INPUT 文の実行のときは、カーソルは表示される。 			
<p>MZ35 CURSOR 10, 10</p> <ul style="list-style-type: none"> ・ 垂直位置は 0 から24、水平位置は 0 から79。 			
<p>XIHu ・ 本文と同じ。ただし、カーソル・スイッチはない。</p> <ul style="list-style-type: none"> ・ CONSOLE で指定した表示エリア外を指定するとエラーになる。 ・ CURSOR も同じ使い方。 			
<p>MB16 ・ カーソル・スイッチは四つのモードがある。0 でカーソルなし。1 で表示、2 で低速ブリンク、3 で高速ブリンク。</p> <ul style="list-style-type: none"> ・ カーソル・スイッチの後に、表示開始行、表示終了行を書ける。範囲は 0 ～7 または、0 ～15。 			
<p>IBM55 LOCATE 10, 10, 0, M1, M2</p> <ul style="list-style-type: none"> ・ 垂直位置は 1 から24の範囲、水平位置は 1 から80の範囲。 ・ M1, M2にてカーソルの大きさを変えることができる。 ・ 各引数はそれぞれ省略できる。 			
<p>PC100 ・ IBM55と同じ。</p>			
<p>MSX ・ 本文と同じ。ただし、水平位置の範囲は画面のモードにより異なる。</p>			

カーソルの縦位置表示 CSRLIN

カーソル・ライン

形式: CSRLIN

機能: CRT のカーソルの現在の垂直位置を与える関数.

特徴: 1 最上位行が 0 である.
2 引数は必要ない.

プログラム例: CRT 上の出力座標を指定し, 数字を出力させ, その水平位置と垂直位置を出力する.

```

10 FOR I=1 TO 5
20 LOCATE 3*I,2*I
30 P1=POS(34)
40 PRINT I;"pos=";P1;"csrlin=";CSRLIN
50 NEXT I

```

実行例

```

1 pos= 3 csrlin= 2
2 pos= 6 csrlin= 4
3 pos= 9 csrlin= 6
4 pos= 12 csrlin= 8
5 pos= 15 csrlin= 10

```

参照: POS, LOCATE

APL —

TRS₁ —

CPM —

M223 RCY
・機能は同じで, $0 \leq RCY \leq 23$.

MZK —

CBM —

PC8 ・本文参照.

TRS_{II} ROW (A)
・Aはダミーで, $0 \leq ROW(0) \leq 23$.PC3 POS X, Y
・変数Yに, 現在の垂直位置を与える.

IF82 —

レベル₃ ・本文と同じ.C180 LPOS
・機能は同じで, $1 \leq LPOS \leq 24$.

M243 ・M223 と同じ.

MZB CSRV
・機能は同じである.BUB ・本文と同じ. ただし,
 $0 \leq CSRLIN \leq 24$.

FM8 ・本文と同じ.

PSP —

PC88 ・本文と同じ.

N52 ・本文と同じ.

PC6 ・本文と同じ.

MLT ・最上位行は 1. 垂直位置
の範囲は 1 ~ 25 までである.

HC ・本文と同じ.

FP11 ・本文と同じ.

SMC ・本文と同じ.

MZ35 ・PC3 と同じ.

X1Hu ・本文と同じ.

MB16 ・MLT と同じ.

IBM55 ・最上位行は 1. 垂直位置
の範囲は 1 から 24 までである.

PC100 ・本文と同じ.

MSX ・本文と同じ.

カーソルの横位置表示 POS

ポジション

形式：POS (A)

機能：CRT のカーソルの現在の横位置 (桁) を与える関数。

特徴： 1 左端が 0 である。
2 引数はデミーである。

プログラム例：CRT 上の出力座標を指定し、数字を出力させ、その水平位置と垂直位置を出力する。34 はデミー。

```
10 FOR I=1 TO 5
20 LOCATE 3*I,2*I
30 P1=POS(34)
40 PRINT I;"pos=";P1;"csrlin=";CSRLIN
50 NEXT I
60 END
```

実行例

```
1 pos= 3 csrlin= 2
2 pos= 6 csrlin= 4
3 pos= 9 csrlin= 6
4 pos= 12 csrlin= 8
5 pos= 15 csrlin= 10
```

参照：CSRLIN, LPOS, LOCATE

APL 0 ≤ POS(0) ≤ 39.

TRS_I 本文と同じ。

CPM 本文と同じ。

M223 本文と同じ。ただし、
CRT の引数は 0 のみである。
・引数に I/O の装置番号を代入すればその装置のヘッドの位置を読み込める。

MZK —

CBM 本文と同じ。

PC8 本文参照。

TRS_{II} 本文と同じ。ただし、
0 ≤ POS(0) ≤ 79.

PC3 POS X, Y
・変数 X に、現在の横位置 (桁) を与える。

IF8₂ 本文と同じ。

レベル₃ 本文と同じ。ただし、引数は 0 のみである。

C180 CPOS
・機能は同じで、1 ≤ CPOS ≤ 80.

M243 本文と同じ。

MZB CSRH
・機能は同じである。

BUB 本文と同じ。

FM8 本文と同じ。

PSp —

PC88 本文と同じ。

N52 POS

PC6 本文と同じ。

MLT 左端が 1 である。

HC A は 0 から 16 までの範囲。

FP11 N52 と同じ。

SMC CSRCLM
・機能は同じ。
・これに関連して、グラフィック座標の位置表示は GPOSX, GPOSY で与えられる。

MZ35 PC3 と同じ。

X1Hu レベル₃ と同じ。

MB16 MLT と同じ。

IBM55 左端は 1。値の最大値は文字モードで 80、グラフィック・モードで 78 である。

PC100 MLT と同じ。

MSX 本文と同じ。

ライト・ペンのエリア指定 PEN

形式: PEN 1; (N1, N2) - (N3, N4)

始点

終点

, 2; (N5, N6) - (N7, N8)

*省略・繰り返し可(8回まで)

機能: CRT画面上を、いくつかのエリアに区分する。エリアは、1から9の番号が付けられる。エリアは、始点、終点を対角の頂点とする長方形である。

- 特徴: 1 水平座標(N1, N3, N5, N7)は、0から79(39)までの値で、垂直座標(N2, N4, N6, N8)は、0から24までの値である。
 2 エリアが重なった場合は、エリア番号の小さいほうが優先される。
 3 ON PEN GOSUB でのエリア番号を定義するのに使う。

プログラム例: CRTにライト・ペン入力エリアを表示し、1から1000までの素数の個数を計算する。計算中にライト・ペンでCRTのTIMEのエリアを押せば、計算時間をプリンタ出力し、PRINTのエリアを押せば、Iの値をプリンタ出力する。

```

100 CLS
110 OPEN "O", #5, "LPT0:"
120 PEN 1; (36, 0) - (39, 1), #2; (36, 3) - (39, 4)
130 LINE (36, 0) - (39, 1), #8, BF
140 LINE (36, 3) - (39, 4), #8, BF
150 LOCATE 30, 0: PRINT "TIME:"
160 LOCATE 30, 3: PRINT "PRINT:"
170 ON PEN GOSUB 320, 300, 310
180 PEN ON
190 TIME #="00:00:00"
200 N=2
210 FOR I=5 TO 1000 STEP 2
220   FOR J=3 TO SQR(I) STEP 2
230     IF I MOD J=0 THEN 250 ELSE NEXT
240   N=N+1
250 NEXT
260 PEN OFF
270 PRINT #5, "素数 (1-1000) の個数: "; N
280 CLOSE
290 END
300 PRINT #5, TIME#: RETURN
310 PRINT #5, "I="; I: RETURN
320 RETURN

```

実行例: (1) (プリンタ出力)

実行例: (2) CRTの表示

```

00:00:06
I= 223
I= 227
00:00:23
I= 567
00:00:31
I= 727
I= 873
00:00:51
227 (1-1000) の素数 168

```

TIME ■
PRINT ■

参照: ON PEN GOSUB, PEN, PEN ON

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 —

IF₈₂ ・本文と同じ。ただし、エリア番号は、1～8、水平座標は0～639、垂直座標は、0～199までの整数値である。

レベル₃ ・本文参照。

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX ・本文と同じ。

ライト・ペン関数 PEN_{関数}

ペン

形式: PEN (N)

N = 0 ~ 5

機能: ライト・ペンの状態を得る。

- 特徴: 1 ライト・ペンからの情報には、トリガ・センスとレベル・センスがある。トリガ・センスとは、ライト・ペンを押した瞬間のもので、レベル・センスとはライト・ペンを押している間の情報である。
- 2 PEN (0) は、トリガ・センスで、ライト・ペンを押すと真(-1)となり、割り込み処理ルーチンへ入るまでその値を保つ。ペンが押されていないと偽(0)となる。
- 3 PEN (1) は、トリガ・センスで、ライト・ペンの水平座標を得る。ただし、 $0 \leq \text{PEN}(1) < 79$ である。
- 4 PEN (2) は、トリガ・センスで、ライト・ペンの垂直座標を得る。ただし $0 \leq \text{PEN}(2) < 24$ である。
- 5 PEN (3) は、レベル・センスで、ライト・ペンを押している間真(-1)となり、押されていないと偽(0)となる。
- 6 PEN (4) は、レベル・センスで、ライト・ペンの水平座標を得る。
- 7 PEN (5) は、レベル・センスで、ライト・ペンの垂直座標を得る。

プログラム例: 主プログラムでは2から1000までの素数を計算し、プリンタに出力するプログラムである。このプログラムを実行中であっても、ライト・ペンで CRT 上に直線を描くことができる(実行例は ON PEN GOSUB を参照)。

```

100 OPEN "0",#5,"LPT0:"
110 COLOR 7,1
120 ON PEN GOSUB 280
130 PEN ON
140 CLS
150 PRINT #5
160 N=2
170 PRINT #5,USING "####";2;3;
180 FOR I=5 TO 1000 STEP 2
190   FOR J=3 TO SQR(I) STEP 2
200     IF I MOD J = 0 THEN 230 ELSE NEXT
210   PRINT #5,USING "####";I;
220   N=N+1:IF N>9 THEN N=0:PRINT #5
230 NEXT I
240 PEN OFF
250 CLOSE
260 END
270 'LPEN ROUTINE
280 ST=ST+1
290 IF ST=1 THEN X=PEN(1):Y=PEN(2) ELSE ST=0
300 LINE (X,Y)-(PEN(1),PEN(2)), "*"
310 RETURN
    
```

参照: ON PEN GOSUB, PEN ON

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・ PEN(0) ~ PEN(5) は、本文と同じ。ただし、 $0 \leq \text{PEN}(1) < 640$ 、で値は、16の倍数である。また、 $0 \leq \text{PEN}(2) < 200$ 、 $\text{PEN}(6)$ 、 $\text{PEN}(7)$ は次のとおり。

・ PEN(6) は、トリガ・センスで、ライト・ペンが押されたときのエリア番号を得る。エリア番号は、PEN 文で指定し、値は $0 \leq \text{PEN}(6) \leq 8$ である。

・ PEN(7) は、レベル・センスで、ライト・ペンが押されたときのエリア番号を得る。また、エリア以外なら 0 となる。

レベル₃ ・ 本文参照。

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 ・ PEN(0) ~ PEN(2) があり。

- ・ PEN(0) は本文と同じ。
- ・ PEN(1) は本文の PEN(4) と同じ。
- ・ PEN(2) は本文の PEN(5) と同じ。

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC ・ PEN は、位置状態が入力されたかどうかを返す。

- ・ PENCLN は PEN(1) と同じ。
- ・ PENLIN は PEN(2) と同じ。
- ・ グラフィック画面の座標の場合、PENX と PENCLN、PENY と PENLIN が同じである。
- ・ PENREAD 命令で、座標を読みとっておくことが必要である。

MZ35 —

XIHu ・ 該当なし。ただし、ジョイスティックが次のように使える。

STICK(0)

- ・ ジョイスティック、テンキーの値を与える。引数は 0 がテンキー、1 はジョイスティック 1、2 はジョイスティック 2。

STRIG(0)

- ・ ジョイスティックのトリガ、スペース・バーの状態を知る。
- 0 はスペース・バー、1 はジョイスティック 1、2 はジョイスティック 2。

ライト・ペン関数 PEN_{関数}

<p>MB16 ・ N は 0 ～ 9 .</p> <p>・ PEN (1) は本文 PEN (3) , PEN (2) はトリガ・センスの水平座標, PEN (3) はトリガ・センスの垂直座標, PEN (4) はレベル・センスの水平座標, PEN (5) はレベル・センスの垂直座標, PEN (6) は本文 PEN (5) , PEN (7) は本文 PEN (4) , PEN (8) は本文 PEN (2) , PEN (9) は本文 PEN (1) .</p>			
<p>IBM55 —</p>			
<p>PC100 —</p>			
<p>MSX ・ 該当なし。ただし、ジョイスティックとパドルとタッチ・パッドが次のように使える。</p> <p>STICK(M)</p> <p>・ ジョイスティックの方向を与える。 M が 0 のとき、カーソル移動キーが対応する。</p> <p>STRIG(M)</p> <p>・ ジョイスティックのトリガ・ボタンの状態を与える。 M が 0 のときスペース・キーが対応する。</p> <p>PDL(M)</p> <p>・ パドルの情報を与える。</p> <p>PAD(M)</p> <p>・ タッチ・パッドの情報を与える。</p>			

マウスの機能設定 MOUSE

マウス

形式：MOUSE 0

MOUSE 1, 0, 0, 1

MOUSE 2, A\$, X\$, 7, 7

MOUSE 3, 0, 100

MOUSE 4, 100, 100, 400, 400

機能：マウスの諸機能を設定する。

- 特徴：1 1 番目の引数の違いにより五つの機能が次のように定められている。
- 0 マウスを初期化してから機能を止める。
 - 1 マウス・カーソルの初期表示位置の設定とスイッチ。
 - 2 マウス・カーソルの形状設定。
 - 3 マウスの移動比率を設定する。
 - 4 マウス・カーソルの移動範囲を指定する。
- 2 マウスはライト・ペンなどと同じく、割り込みルーチンを定めてから用いる MOUSE (A1) ON, MOUSE (A1) OFF, MOUSE (A1) STOP がある。
- 3 マウスの諸状態を返す関数として次のものがある。
- (a) MOUSE (A1) : A1=0 では現在のマウスの X 座標, A1=1 では現在のマウスの Y 座標, A1=9 ではこの関数が次に呼び出されたときまでの移動した X 方向の距離, A1=10 は同じく Y 方向の距離。
 - (b) MOUSE (A2, A3) : A3 はボタン番号で, A2 は 2~8 の値がはいり各々ボタンが押されたときの状態が得られる。

プログラム例：マウスの機能設定をし、マウスの位置を検出して出力する。

```

100 MOUSE 0:MOUSE 1,,0:MOUSE 1.360,256,1
110 CLS
120 MOUSE <2> ON
130 ON MOUSE<2> GOSUB 160
140 PRINT X,Y:LOCATE 1,1
150 GOTO 130
160 X=MOUSE<0>:Y=MOUSE<1>:RETURN
実行例
208      367
344      440
571      398

```

参照：

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1H_u —

MB16 —

IBM55 —

PC100 ・本文参照。

MSX —

グラフィック画面のモード設定 SCREENモード

スクリーン

形式: SCREEN M1, M2, M3, M4

省略可

機能: グラフィック画面に対して種々のモードを設定する。

- 特徴: 1 M1で画面モードを指定する。M1の値は0から2で、0はカラー・モード(640×200ドット)、1は白黒モード(640×200ドット、3ページ)、2は高分解能白黒モード(640×400ドット)。
- 2 M2で画面スイッチを指定する。M2の値は0から3で、0は高速書き込み、グラフィック・マスク・スイッチ共にOFF、1は高速書き込みスイッチのみON、2はグラフィック・マスク・スイッチのみON、3は高速書き込み、グラフィック・マスク・スイッチ共にON。
- 3 M3ではアクティブ・ページ、M4ではディスプレイ・ページを指定するが、これは白黒3ページ・モード(M1=1のとき)に指定する。
- 4 M3はグラフィックで書き込むページで、0から2で指定する。
- 5 M4は3ページのうちのどのページを表示するかを示すもので、0から7の値が入る。0は全ページ表示しない、1はページ1のみ表示、2はページ2のみ表示、3はページ1とページ2を合成して表示、4はページ3のみ表示、5はページ1とページ3を合成して表示、6はページ2とページ3を合成して表示、7は全ページを合成して表示。

プログラム例: VIEW, WINDOW を参照。

参照: VIEW, WINDOW

APL ・GRでテキスト・モード。
・HGRで高分解能モード。TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ SCREEN M1, M4, N
 ・M1はグラフィック・モードで、0でノーマル、1で高解像度。
 ・M4は表示画面の指定。
 ・Nは0でインターレス・モード、1でノンインターレス・モード。

C180 —

M243 —

MZB SCREEN M1, M2, M4
 ・M1は1~2で、1はカラー・モード、2は白黒モード。
 ・M2はグラフィック・モードで、1でノーマル、2で高解像度。
 ・M4は高解像度の表示を指定する。

BUB —

FM8 —

PSP SCREEN M
 ・Mは0~2で、0はテキスト・モ

ード、1はグラフィック・モード、2はファイン・グラフィック・モード。

PC88 ・本文参照。

N52 —

PC6 SCREEN M1, M3, M4
 ・画面スイッチの指定はない。
 ・M1は1から4で画面モードを指定する。
 ・M3は1から4で有効になるページを指定する。
 ・M4は1から4で表示されるページを指定する。

MLT —

HC SCREEN M, M1
 ・Mは0と1で、0はLCDに表示、1は外部ディスプレイに表示。
 ・M1は0から2で、0はLCD表示、1は外部ディスプレイ表示、2は外部ディスプレイの高分解能モードに指定する。

FP11 SCREEN M1, M4, M3, M5
 ・M1は0から2、M4は0から7、M3は0から7で指定する。
 ・M4, M3はM1が2(カラー・モードRGB三画面の独立表示・合成表示)のとき有効。
 ・M5はホワイト・オプションで0か1、1にすると白黒でも三画面の独立表示・合成表示が可能。

SMC ・該当なし。ただし、GMode M1, M6で画面モードを、GPLANE A1, A2でページ数の選択を行う。
 ・M1は0から3で、数が大きいほど分解能が高い。
 ・M6はM1が2の時の表示カラーの

グラフィック画面のモード設定 SCREENモード

スクリーン

<p>指定で0か1,あるいは省略で、省略のとき0になる。</p>	<p>IBM55 SCREEN M ・Mは0から2で、0は文字モード1または2がグラフィック・モード。</p>		
<p>MZ35 ・該当なし。ただし、ODISP M3で書き込む画面の指定を、SDISP M4で表示する画面の指定を行う。 ・M3は1から3,あるいはこの3文字の組み合わせである。 ・M4は0から3,あるいはこの4文字の組み合わせである。</p>	<p>PC100 ・M1は0と1のみ使用可、M2は縦横スイッチで横は0, M3はキャラクタ・モード・スイッチ, M4はカナ漢字変換スイッチ。</p>		
<p>X1Hu SCREEN M1, M2, M3 ・M1で出力ページ, M2で入力ページ, M3でグラフィック・モードを指定する。 ・GRAPHも同義。以下のような関連する命令がある。 OPTION SCREEN 2 ・グラフィック用メモリを外部記憶に用いる。1のときグラフィック・メモリ。 PRW &B00000001 ・ビットに対応する色のテキストとグラフィック画面の優先を決める。ビット番号に対して色が対応し、1のときグラフィックが優先される。 LAYER 2, 1, 3, 4 ・テキストとグラフィック画面1, 2, 3の優先順位を決める。</p>	<p>MSX SCREEN M1, M2, M3, M4, M5 ・M1は画面モードで、0~3, M2はスプライト・サイズで0~3, M3はキー・クリック・スイッチで0か1(1で音出し), M4はボーレートで1か2(1で1200ボー, 2で2400ボー), M5はプリンタのタイプ(0以外で, MSXのキャラクタ・セットをもたないプリンタ)。</p>		
<p>MB16 SCREEN M1, M2, M3, M4, M5, M6, M7 ・M1は画面モード, 0~4の範囲, M2は0または正の値をもつ数式, M3はアクティブ・テキスト・ページ, M4はディスプレイ・テキスト・ページ, M5はインターレースか, ノンインターレース・モード設定で0~3の範囲, M6はアクティブ・グラフィック・ページ, M7はディスプレイ・グラフィック・ページ。</p>			

画面領域設定 VIEW, WINDOW

ビュー, ウインド

APL —	・実画面の左上の隅が、仮想スクリーンの上で水平座標、垂直座標の位置になるように、実画面を移動する。	PC100 ・VIEW に関しては同じ。 ・WINDOW は WINDOW ^{省略可} [SCREEN]
TRS _I —		(X1, Y1) - (X2, Y2) のように用いられ、SCREEN はオプション。
CPM —		・オプションの SCREEN は、付けると上から下に向かって y 座標が増加するような座標系になる。
M223 —	FP11 ・機能の②はない。形式は同じ。ただし、色の指定はない。 INIT (320, 100), 1, 1, R ・絶対座標に対するユーザ座標を設定する。	
MZK —	・中心座標、X 増分、Y 増分の順に指定。文の最後に、R をつけると、タテ・ヨコの比を 1 : 1 にする。 SCALE (-1, 1) - (1, -1)	
CBM —	・VIEW 文で指定されたグラフィック表示領域の左上隅、右下隅の座標を設定する。	MSX —
PC8 —	・新たに VIEW 文が実行されても、SCALE は自動的に変化しない。	
TRS _{II} —		
PC3 —		
IF8 ₂ —		
レベル ₃ —	SMC GLOCATE (10, 10), 2, -2 ・座標の原点を設定する。	
C180 —	・中心座標は、-16384 から 16383 までの範囲で指定する。	
M243 —	・第 2 の引数は横軸、第 3 の引数は縦軸の方向を指定する。	
MZB —		
BUB —	MZ35 ・該当なし。ただし、原点の位置および座標の大きさ（スケール）を設定するための SCALE 命令がある。	
FM8 —		
PSP —		
PC88 ・本文参照。	X1Hu WINDOW (0, 0) - (100, 50), (-1000, -1000) - (1000, 0) ・() 単位で順に絶対座標系の左上頂点、右下頂点、論理座標系の左上頂点、右下頂点となる。	
N52 —	・論理座標を省略すると、絶対座標系となる。	
PC6 —		
MLT —		
HC LOCATES 0, 0 ・第 1 の指定は水平座標、第 2 の指定は垂直座標。	MB16 — IBM55 —	

座標の変換 PMAP

ピー・マップ

形式	S=PMAP (W, 1)
機能	式Wで示されるワールド座標,あるいはスクリーン座標を2番目の引数によって指定される機能によって,スクリーン座標あるいはワールド座標に変換する.
特徴	1 機能指定は, <ul style="list-style-type: none"> 0: Wをワールド座標 (X座標) として,スクリーン座標に変換する. 1: Wをワールド座標 (Y座標) として,スクリーン座標に変換する. 2: Wをスクリーン座標 (X座標) として,ワールド座標に変換する. 3: Wをスクリーン座標 (Y座標) として,ワールド座標に変換する.
プログラム例	<p>ワールド座標 (X,Y) からスクリーン座標 (SX, SY) への変換とその逆を行う.</p> <pre> 100 WINDOW <-100.100>-<100.100> 110 DEFINT S=2 120 X=10:Y=10:PRINT "SX=" ;PMAP(X,0) ."SY=" ;PMAP(Y,1) 130 X=5:Y=5:PRINT "WX=" ;PMAP(X,2) ."WY=" ;PMAP(Y,3) </pre> <p>実行例</p> <pre> SX= 395 SY= 230 WX=-98.60918 WY= 98.042 </pre>
参照	

APL	—
TRS _I	—
CPM	—
M223	—
MZK	—
CBM	—
PC8	—
TRS _{II}	—
PC3	—
IF8 ₂	—
レベル ₃	—
C180	—
M243	—
MZB	—
BUB	—
FM8	—
PSP	—
PC88	—
N52	—
PC6	—
MLT	—
HC	—
FP11	—

SMC	—
MZ35	—
X1Hu	—
MB16	—
IBM55	—
PC100	・ 本文参照.
MSX	—

色指定 COLOR

カラー

形式: COLOR 4, 0, 0

□ □ □

省略可

機能: 画面の色, 機能を指定する。

特徴: 1 白黒モードでは, 第1の引数は画面のモードを指定する。
 カラー・モードでは, 第1の引数は以下のように色を指定する。

0	1	2	3	4	5	6	7
黒	青	赤	紫	緑	水色	黄	白

- 2 第2の引数は, 画面をクリアした時のキャラクタ・コードを指定する。通常は, ナル・キャラクタ・コード0を指定(何も表示しない)。
 3 第3の引数は, 1のときグラフィック・モードに, 0のときはキャラクタ・モードになる。
 4 これらの引数は省略すると前の状態を保つ。
 5 白黒モード/カラー・モードは, CONSOLE コマンドで指定する。

プログラム例: カラー・モードで COLOR 0~7 をその色で画面に出力する。

```
10 CONSOLE 0,25,0,1
20 FOR I=0 TO 7
30 COLOR I
40 PRINT"COLOR ";I
50 NEXT I
60 END
```

実行例

```
COLOR 0 ----> 黒
COLOR 1 ----> 青
COLOR 2 ----> 赤
COLOR 3 ----> 紫
COLOR 4 ----> 緑
COLOR 5 ----> 水色
COLOR 6 ----> 黄
COLOR 7 ----> 白
```

APL HCOLOR=5 または,
 COLOR=5

- ・前者は高解像モードのときに使い, 引数は0~7まで8種の色表示を行う。
- ・後者は低解像モードのときに使い, 引数は0~255までとり, 16のモードで16種類の色表示を行う。

TRS_I —

CPM —

M223 —

M2K —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ COLOR 7, 2, 0, 2

- ・第1の引数で文字色, 第2の引数で画面の背景色, 第3の引数で画面のわくの色を指定する。第4の引数で色コードを指定すると, 画面上の色コードとの AND をとったものを表示する。

レベル₃ COLOR 7, 1

- ・第1の引数で文字色, 第2の引数で背景色を指定する。どちらの引数も省略できる。
- ・COL. という省略形も使える。

C180 —

M243 —

M2B —

BUB ・レベル₃と同じ。ただし, 省略形は使えない。

FM8 ・レベル₃と同じ。

PSP ・BUB と同じ。

PC88 COLOR @ (0, 10)-(50, 60), 3

- ・二つの座標間の直線を対角線とする四角形の内部に対して, 次の引数で色, 機能を指定する。省略すると7に設定される。
- ・座標はキャラクタ・モードで使う。
- ・これに関連して, COLOR=(M1, M2) によりグラフィック画面のパレットの変更が可。

N52 COLOR アトリビュート・コード

- ・アトリビュート・コードでカラー, アンダー/オーバーライン, リバースなどの機能を指定する。

PC6 COLOR 4, 1, 1

- ・第1の引数で文字色, 第2の引数で背景色, 第3の引数でカラー・セットを設定する。

MLT ・レベル₃と同じ。ただし, 省略形は使えない。

HC ・PC6と同じ。

FP11 COLOR 4, 7, 3, 6

- ・指定は, 文字色, 背景色, グラフィック色, 表示外色の順, いずれも0から7まで。

SMC CCOLOR M1, M2, N1

- ・キャラクタ表示の色を指定する。
- ・M1, M2 は-32768から32767までの範囲で, N1 は0から255までの範囲をとる。
- ・M1はキャラクタ・カラー, M2は

参照: CONSOLE

色指定 COLOR

<p>バック・グラウンド・カラー, N1は属性を指定する。</p> <ul style="list-style-type: none"> 省略すると、それ以前のCCOLOR文で指定した値を保持する。 <p>GCOLOR M1, M2, M3, N2</p> <ul style="list-style-type: none"> グラフィック表示の色を指定する。 M1, M2, M3は-32768から32767までの範囲で、N2は0から4までの範囲をとる。 M1は表示カラー、M2はバック・グラウンド・カラー、M3はボーダー・エリア・カラー、N2は論理演算コードを指定する。 省略すると、それ以前のGCOLOR文で指定した値を保持する。 <p>KCOLOR M1, M2, N1</p> <ul style="list-style-type: none"> グラフィック表示上の漢字の色を指定する。 M1, M2は-32768から32767までの範囲で、N1は0から255までの範囲をとる。 M1は漢字カラー、M2はバック・グラウンド・カラー、N1は属性を指定する。 省略すると、それ以前のKCOLOR文で指定した値を保持する。 <p>ACLEAR 5, 5, 3, 1</p> <ul style="list-style-type: none"> キャラクタ表示面上の、指定した範囲に表示されている文字のアトリビュートを変更する。 第1の引数は変更開始行、第2の引数は変更行数を指定する。 第3の引数はキャラクタ・カラー、第4の引数はバック・グラウンド・カラー、第5の引数は属性コードを指定する。 実行後、表示される文字には影響がない。 第1の引数を省略すると最上行が指定され、第2の引数を省略すると最下行までが指定される。 第3, 4, 5の引数を省略すると、それ以前に指定されたままとなる。 <p>MZ35 CHANGE DISP S</p>	<p>・文字定数Sの示す内容により、文字表示モードの設定およびモノクロ/カラー・モードの選択を行う。</p> <ul style="list-style-type: none"> 電源ON時には、80×25モノクロ・モードが自動的に設定される。 <p>COLOR</p> <ul style="list-style-type: none"> 文字(擬似グラフィック・シンボルを含む)の色を指定する。 <p>GCOLOR 5, 3</p> <ul style="list-style-type: none"> グラフィック画面における、グラフィック・パターンと背景の色を指定する。 <p>BL 0, 2, 80</p> <ul style="list-style-type: none"> CRT表示の文字をブリンク(点滅)状態にする。 <p>RV 0, 2, 80</p> <ul style="list-style-type: none"> CRT表示の文字をリバース(反転)状態にする。 <p>REBL 0, 2, 80</p> <ul style="list-style-type: none"> DISP CLEAR文でクリアできないブリンク、リバース状態をクリアする。 <p>XIHu COLOR 7, 0</p> <ul style="list-style-type: none"> 第1の引数は表示色、第2の引数は背景色で色番号は本文と同じ。 <p>CANVAS 1, 2, 3</p> <ul style="list-style-type: none"> マルチ画面の各グラフィック画面の色を決める。 <p>PALET 4, 0</p> <ul style="list-style-type: none"> パレットの色を決める。 <p>MB16 COLOR M1, M2, M3, M4</p> <ul style="list-style-type: none"> M1は文字色、M2は背景色、M3は境界色で0～15の範囲。 <p>IBM55 COLOR M1, M2, M3</p> <ul style="list-style-type: none"> Dは機能を指定する0～255の範囲の数式。 パレット変更には PALETTE 文で行う。 本文とは意味が異なり、文字の下線づけ、反転、罫線の指定をする。 <p>PC100</p> <ul style="list-style-type: none"> 同名の命令だが意味は大 	<p>きく違う。</p> <ul style="list-style-type: none"> COLOR M1, M2の形で使用され、M1はフォア・グラウンド・カラーで、キャラクタ、点、線はこの色で表され、M2はバック・グラウンド・カラーで、画面の下地の色を表す。 PC100では、パレットの概念を用いているため、通常のCOLOR命令に相当するものとして PALETTE や PALETTE USINGがある。 <p>MSX COLOR M1, M2, M3</p> <ul style="list-style-type: none"> M1はフォア・グラウンド・カラーで、文字、色指定なしのグラフィックがこの色で表される。 M2は背景色、M3は境界色、すべて0～15まで。 	
--	---	--	--

線引き LINE (その1)

ライン

形式: LINE 6, 2

機能: 1 行単位での画面モードの指定をする。

- 特徴:
- 1 カラー・モードでのみ使う。白黒モードではエラーになる。
 - 2 第一の引数は、0 から行数-1 までの値をとり、画面の行指定をする。
 - 3 第二の引数は、ファンクション・コードで0~3 までの値をとり、0:ノーマル、1:ブリンク、2:リバース、3:リバース・ブリンクとなる。

プログラム例: 1 行目をリバース・モードにして出力する。

```
10  CONSOLE 0,25,0,1
20  PRINT CHR$(12)
30  LOCATE0,0:PRINT"***  TEST  PROGRAM  ***"
40  LINE 0,2
50  END
```

実行例

```
***  TEST  PROGRAM  ***
```

参照: CONSOLE, COLOR

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 ・該当なし。ただし、REV.

NORM 命令で、表示の反転、解除ができる。

SMC —

MZ35 —

X1Hu ・該当なし。ただし、1 文字単位で CREV, CFLASH 命令によって反転、点滅することができる。

MB16 —

IBM55 ・該当なし。ただし、COLOR 命令により、文字の反転ができる。

PC100 —

MSX —

線引き LINE (その2)

形式: LINE (5, 5) - (20, 10), "A", 4, B

省略可

機能: キャラクタで線分や四角形を描く。

- 特徴: 1 最後に B (あるいは BF) を付けしないと, キャラクタを使って
初めの座標 (X1, Y1) から次の座標 (X2, Y2) まで線を描く。
水平座標 X1, X2 は, 0 から指数-1 まで, 垂直座標 Y1, Y2
は, 0 から行数-1 までの値をとる。
- 2 第2の指定で, 引用符 (") で囲まれた文字列の最初の一文字
のみをキャラクタとする。
- 3 第3の指定 (0 ~ 7 までの数字) により, 色, 機能を決めること
ができる。これは COLOR 文のファンクション・コードと同じ。
- 4 最後に, B を付けると, (X1, Y1), (X2, Y2) 間の直線を対角
線とする四角形をキャラクタを使って描く。BF のときは, そ
の四角形の内部をすべて指定キャラクタで埋める。

プログラム例: 座標 (2, 10) と (17, 7) の間の直線を対角線とする四
角形を指定文字で出力する。

```
10 PRINT"START POSITION(x,y)=(2,10)"
20 PRINT"END POSITION(x,y)=(17,7)"
30 INPUT"CHARACTER";A$
40 LINE(2,10)-(17,7),A$,B
50 END
```

実行例

```
START POSITION(x,y)=(2,10)
END POSITION(x,y)=(17,7)
CHARACTER? M
```

```
MMMMMMMMMMMMMMMMM
M                      M
M                      M
M                      M
MMMMMMMMMMMMMMMMMMMM
```

参照: LINE (その3)

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ ・本文と同じ。ただし, X,
Y 座標はそれぞれ 0 ~ 39 (79),
0 ~ 24 までである。

C180 —

M243 —

MZB —

BUB —

FM8 LINE @ (5, 5) - (20, 10),
"A", 4, B
・機能は本文と同じ。

PSP ・本文と同じ。

PC88 —

N52 —

PC6 LINE (X1, Y1) - (X2,
Y2) [色], [B(F)]
・キャラクタで描けない。

MLT —

HC —

FPI1 —

SMC —

MZ35 ・該当なし。
・表作成のための格子を描くための
TABLE 命令がある。

X1Hu ・本文と同じ。ただし, 第
3 引数はない。

MB16 —

IBM55 —

PC100 —

MSX —

線引き LINE (その3)

ライン

形式: LINE (0, 0) - (50, 50), PSET, 4, B

または PRESET 省略可

機能: ドットで線分や四角形を描いたり、消したりする。

- 特徴: 1 初めの座標 (X1, Y1) から、次の座標 (X2, Y2) まで、第2の指定が PSET のときは線を描き、PRESET のときは線を消す。水平座標 X1, X2 は 0 ~ (行数 * 2 - 1)、垂直座標の Y1, Y2 は 0 ~ (行数 * 4 - 1) までである。
- 2 第3の指定は、COLOR 文のファンクション・コードと同じで、色、機能を示す。
- 3 第4の指定は、B のとき四角形を描き、BF のときドットで四角形の中まで埋める。省略すると線分を描く。

プログラム例: LINE 命令で箱を使って、画面にパターンを描く。

```
10 FOR I=10 TO 2 STEP -2
20 IX=I*2: IY=I*2
30 LINE (IX, IY)-(26-IX, 30-IY), PSET, B
40 NEXT
50 LOCATE 0, 7: END
```

実行例



参照: LINE (その2), PSET, PRESET, COLOR

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ ・第3の引数で色を指定でき、PRESET のときも指定できるが、通常は背景色と同じにする。また初めの座標は省略でき、そのとき (0, 0) に設定される。

レベル₃ ・IF8₂ と同じ。X, Y 座標は、それぞれ 0 ~ 639, 0 ~ 199, 色コードは 0 ~ 15 まで指定できる。

C180 ・該当なし。ただし、表を表示するとき便利な SETFORM がある。ACCEPT, DISPLAY を用いると表内容の入出力ができる。

M243 —

MZB LINE 50, 210, 50, 150
BLINE 50, 210, 50, 150

・LINE で指定座標を次々とドットでつなぎ、BLINE で同様に直線を描く。

・引数の奇数番目が水平座標、偶数番目が垂直座標を示す。

BUB LINE@ (0, 0) - (50, 50),

1, B

- ・@ は省略可、線を描くのみ。
- ・第1の座標を省略すると、その前の LINE 実行の際、第2の座標に設定される。初めは、(0, 0) に設定されている。
- ・第2の指定で色コードは 0 ~ 7 まで指定できる。ほかはレベル₃ と同じ。

FM8 LINE (0, 0) - (50, 50),
AND, 2, B

・第2の指定で、PSET で線を描き、PRESET で線を消す。AND, OR, XOR のとき、第3の指定の色コードと画面の色との論理演算が行える。ほかはレベル₃ と同じ。

・これに関連して、多数個の座標間を次々に直線で結ぶ CONNECT があ

PSP —

PC88 LINE (0, 0) - (50, 50),
7, &H9F9F

・水平、垂直座標とも -32768 ~ 32768 までである。

・座標の前に STEP を付けると、相対座標も使える。

・第1の座標を省略すると、最後に指定された座標に設定される。

・第3の指定は色を示す。

・第4の指定で、B または BF を指定できる (機能は本文と同じ)。

また、線のパターンも指定でき、通常は16進数で指定する。

N52 —

PC6 —

MLT LINE [STEP] (0, 0) -
[STEP] (50, 50), 1, BF

・第3の引数で線の色を指定し、第4の引数で、B または BF を指定

線引き LINE (その3)

<p>できる。</p> <ul style="list-style-type: none"> ・(STEP)を指定すると相対位置を指定したことになる。 ・最初のSTEP指定は、直前に終了した図形の、終了点座標からの相対位置。省略可。 ・あとのSTEP指定は、始点からの相対位置。 	<p>最後に GCOLOR 文で指定した値になり、GCOLOR 文がない時は、表示カラーは 7、論理演算は 0 になる。</p> <p>BOXE (X1, Y1) - (X2, Y2), 5, 4</p> <ul style="list-style-type: none"> ・四角形を描く。 ・指定は LINE 文と同じ。 <p>BOXF (X1, Y1) - (X2, Y2), 5, 4</p> <ul style="list-style-type: none"> ・四角形を描き、その中を塗りつぶす。 ・指定は LINE 文と同じ。 ・DRAW 文でも同様のことが簡単にできる。 	<p>な座標。</p> <ul style="list-style-type: none"> ・カラー・コードは 0 ~ 7。 ・DRAW "コマンド列" によりスクリーン上に直線図形が描ける。 <hr/> <p>IBM55 ・MB16 と同じだが、色指定は 0 か 1 である。</p> <hr/> <p>PC100 ・PC88 と同じ。</p> <ul style="list-style-type: none"> ・同様の機能で DRAW がある。これは LINE 文の第 1 引数を省略した場合と似た機能をもつ。 <hr/> <p>MSX ・MB16 と同じ。ただし、カラー・コードは 0 ~ 15。</p>	
<p>HC ・四角形は描けない。また、特徴 3 はない。</p> <hr/> <p>FP11 DRAW = (&HF0F0) (0, 0) - (50, 50)</p> <ul style="list-style-type: none"> ・指定はライン・タイプ、座標の順。ライン・タイプは省略可。座標はユーザ座標。 ・座標間に (,) を使うと点が表示され、(-) を使うと、その間を直線で結ぶ。 ・ライン・タイプの指定により、ビット・パターンを使って点線などを作図できる。 ・DRAW 文を使うと背景色で表示する。 <p>PLOT (0, 0) - (50, 50), 6, AND</p> <ul style="list-style-type: none"> ・PLOT 文でも同じことができる。ただし、こちらには、色と機能の指定が付く。 <p>QUAD (0, 0) - (50, 50), 6, AND</p> <ul style="list-style-type: none"> ・QUAD 文を使うと四角形が描ける。 	<p>MZ35 LINE 5, 5; 20, 10, "A", 3, N</p> <ul style="list-style-type: none"> ・第 1, 2 の引数は線の始めの座標を指定し、第 3, 4 の引数は線の終わりの座標を指定する。 ・第 3 の指定で直線の種数を指定し、第 5 の引数で色コードを指定する。 ・第 5 の指定で、N は始点と終点を結び、F は始点と終点を結ばない。 ・LINERESET 5, 5; 20, 10 で直線を消せる。 <hr/> <p>X1Hu LINE(0, 0) - (50, 50), PSET, C, B, &HFFFF</p> <ul style="list-style-type: none"> ・第 1 の座標を省略すると、最後に指定された座標に設定される。第 3 の指定は PSET, PRESET, XOR がある。第 4 の指定でバレット・コード(省略可)、第 5 の指定で B または BF (省略可)、第 6 の指定で線のパターン(省略可)、BF 指定時はタイル・パターン。 		
<p>SMC LINE (X1, Y1) - (X2, Y2), 5, 4</p> <ul style="list-style-type: none"> ・第 2 の指定はカラー・コード、第 3 の指定は論理演算コードを指定する。 ・始点座標を省略すると、直前のグラフィック・コマンド中で最後に指定した座標、または DRAW 文の現在地になる。どちらでもないときは (0, 0) になる。 ・第 2 または第 3 の指定がない時は 	<p>MB16 LINE (0, 0) - (50, 50), 4, B</p> <ul style="list-style-type: none"> ・PSET, PRESET はない。 ・座標の前に STEP を付けると、相対座標も使える。 ・第 1 座標を略すと最後に指定され 		

楕円の表示 CIRCLE

サークル

形式: CIRCLE (80, 80), 40, 2, 0, 6.28, 0.8

中心座標

半径 ↑ 開始 終了 長短の比
角度 角度
ハレット
番号

機能: 指定された場所に指定された大きさ、色、比の楕円の全部または一部を描く。

- 特徴:**
- 1 中心座標は、ワールド座標¹⁾で示す。相対座標²⁾で STEP (10, 12) のように指定してもよい。
 - 2 ハレット番号は、楕円の色を指定する。ハレット番号³⁾と色の対応は、COLOR 文で指定変更できる。その対応が変更されると、上の楕円の色も自動的に変わる。ハレット番号を省略すると、COLOR 文で指定したフォア・グラウンド・カラーになる。
 - 3 開始角度と終了角度を指定することによって、部分弧を描くことができる。-2 π から2 π の範囲で指定する。負の角度は絶対値化がとられるが、そのとき、中心から周まで線が描かれる。省略値はそれぞれ0と2 π 。
 - 4 長短の比は (垂直方向の径) / (水平方向の径) で示される。そのとき半径は、垂直と水平のそれぞれの径のうち、長い方の値をあらわす。比を1とすると円になる。
 - 5 円や楕円や弧や扇形を描かせるのに用いる。

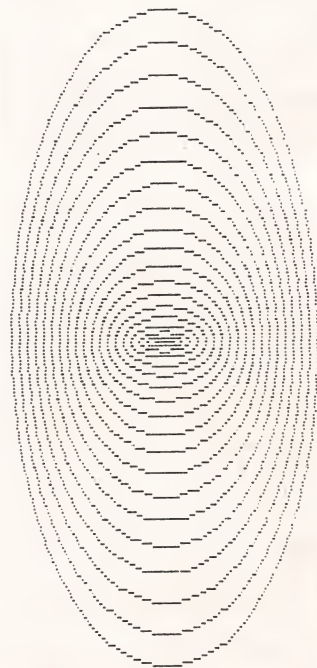
用語: ¹⁾ワールド座標 ユーザが使うことのできる最大の座標系であり、縦・横とも -32768 から 32767 まで許されており、負の領域座標も表現することができる。

²⁾相対座標 座標系内のある1点を基点とした時、X方向、Y方向にどれだけ相対的に移動するかによって決められる座標。

³⁾ハレット番号 ハレットとは、その名の示すとおり絵を描く時に使うハレットという意味で、0番から7番までの八つのハレットをもち、ユーザはこれらのハレットに好みの色を設定することができる。

プログラム例: パラメータを種々に変えて楕円を描く。

```
10 SCREEN 0,0
20 CLS 3
30 FOR I=1 TO 100 STEP 5
40 CIRCLE(300,100),I,6,,,I/100
50 NEXT I
60 END
```



参照: COLOR

楕円の表示 CIRCLE

サークル

APL —	2, 0.8, 0, 1, F, PSET	GCOLOR 文で指定したコードになり、GCOLOR 文がなければ、表示カラーは白で、論理演算はなしになる。
TRS _I —	・機能は BUB と同じであるが、F によって円内の塗りつぶしができる (N なら塗りつぶさない)。	・DRAW 文でも同様なことが簡単にできる。
CPM —	PSET は追加された機能で、PRESET, AND, OR, XOR も使える。	
M223 —		
MZK —	PSP —	MZ35 CIRCLE 80, 80, 40, 0.8 : 0, 360, 5, N
CBM —	PC88 ・本文参照。	・第 1 と 2 の引数で中心点の座標を指定する。第 3 の引数で長軸の長さを指定し、第 4 の引数で短軸と長軸の比率を指定する。第 5 の引数は開始角、第 6 の引数は終了角を指定する。第 7 の引数で表示色を指定し、第 8 のキャラクタで表示形式を指定する。
PC8 —	N52 —	
TRS _{II} —	PC6 ・本文と同じ。	X1Hu CIRCLE (100, 100), 50, 2, 1, 0, 360
PC3 —	MLT ・CIRCLE (STEP) (80, 80), 40, 2, 0, 6.28, 0.8 で、STEP を指定すると、中心点の座標が、直前の図形表示終点からの相対位置で与えられる。	・第 4 引数までは本文と同じで、バレット番号の次に扁平率がくる。
IF8 ₂ CIRCLE(80, 80), 40, 2, 0.8, 0, 1	HC —	・角度は 360 度表現。
・形式は本文と同じであるが、長短の比は後から 3 番目の引数で示す。	FP11 CIRCLE (80, 80), 40, 2, AND	POLY (100, 100), 50, 1, 144, 90, 810
・長短の比は (水平方向の半径のドット間距離) / (垂直方向の半径のドット間距離)、0 ~ 1 の範囲のみ。	・角度、長短の比の指定はない。	・多角形を描く。
・省略すると 0.4177。このとき水平方向の径が半径に一致する。	AND は機能指定で、ほかに、OR, XOR, NOT, OFF がある。	・第 4 引数までは本文と同じで、バレット番号の次にステップ角、初期角、終了角を与える。
・開始角度と終了角度は、円を 8 等分した 0 ~ 1 の値のみ。	・中心座標、半径は共にユーザ座標で示す。	
・時計方向に動く。	SMC CIRCLE (80, 80), 20, 10, 4, 3	MB16 ・本文と同じ。ただし、中心座標、STEP により相対座標とする。
レベル ₃ —	・第 1 引数、第 2 引数は中心座標を指定する。第 3 引数は X 軸半径で、第 4 引数は使用半径を指定する。	IBM55 ・MLT と同じ。ただし、バレット番号は 0 か 1 である。
C180 —	第 5 引数は表示カラー・コードで、第 6 引数は論理演算コードを指定する。	PC100 ・MLT と同じ。
M243 —	・第 3 または第 4 の引数のいずれかを省略すると、指定されている軸方向の数値を半径とする真円を描く。第 5 または第 6 の引数を省略した時は、最後に実行した	MSX ・MLT と同じ。ただし、バレット番号は 0 ~ 15 までのカラー・コード。
MZB —		
BUB ・機能は IF8 ₂ と同じであるが、開始角度、終了角度は 8 等分値でなくてよい。		
FM8 CIRCLE@ (80, 80), 40,		

塗りつぶし PAINT

ペイント

形式：① PAINT (80, 80), 3, 2

座標

領域色 ↑
境界色

② PAINT (80, 80), TILE\$, 2, BACK\$

座標

タイル・ス
tring↑
バック・グラウ
ンド・ストリン
グ
境界色

機能：①画面上のある領域を、色で塗りつぶす。

②画面上のある領域を、模様のパターンで埋めつくす。

特徴：①(1)座標は、ワールド座標 (CIRCLE の項参照) で与えるが、相対座標で STEP (10, 12) のように与えてもよい、この座標の点を含むある領域を塗りつぶす。

(2)領域色、境界色ともにパレット番号で指定する。省略は、フォア・グラウンド・カラー¹⁾である。領域色によって塗りつぶしが行われる。塗りつぶしの範囲は、境界色で指定される。すなわち、境界色で囲まれていて、指定の座標を含む領域を塗りつぶす。

②(1)境界色の意味は①と同じ。タイル・ストリングで指定された模様によって領域が埋められる。

(2)バック・グラウンド・ストリングは、模様を描く前に描かれている模様のパターンで、省略すると COLOR 文で指定されたバック・グラウンド・カラー²⁾になる。

(3)座標は①と同じ。

(4)模様は、ドットのパターンと、文字のコードのビット列を対応させて表す。例えば、白黒のとき次のようになる。

●○○●●●●●→16進数の 9F

●●○○○○●●→16進数の C1

●●●●●●●●→16進数の EF

よって上の模様は、TILE\$ = CHR\$ (&H9F) + CHR\$ (&HC1) + CHR\$ (&HEF) なるタイル・ストリングで表せる。縦に何行か重なってもよい。横は 8 ドットである。カラーの場合は 3 文字で 1 行を表し、第 1 の文字が青、第 2 の文字が赤、第 3 の文字が緑を表す。

用語：¹⁾フォア・グラウンド・カラー グラフィック画面に点や線を表示したりする時に使われる色。²⁾バック・グラウンド・カラー グラフィック画面における背景の色。

プログラム例：円を描かせその中をカラーで塗りつぶす。

```

10 SCREEN 0,0
20 CLS 3
30 FOR I=1 TO 50
40 C=INT(6*RND+1.9)
50 X=639*RND:Y=199*RND
60 CIRCLE(X,Y),RND*80,C
70 PAINT(X,Y),C
80 NEXT I
90 END

```

実行例



参照：COLOR

塗りつぶし PAINT

ペイント

APL —	3, 2 ・機能の②はない。
TRS _i —	・STEPは相対位置の指定を行う時に使う。この時の塗りつぶし開始点は、直前に終了した図形の終了点座標からの相対位置。STEPを省略した場合、絶対座標。
CPM —	・第2の指定は領域色、第3の指定は境界色。
M223 —	
MZK —	
CBM —	HC —
PC8 —	FP11 ・IF8 ₂ と同じ。
TRS _{II} —	SMC ・IF8 ₂ と同じ。
PC3 —	MZ35 PAINT X, Y, Z, U; V ・Xは塗り始める点の横座標、Yは塗り始める点の縦座標を指定する。 Zは塗りつぶす色、Uは境界の色、Vは色の濃淡を指定する。
IF8 ₂ ・本文の1の機能のみあり、形式は同じである。 ・ワールド座標や相対座標は使えない。	X1Hu ・本文の①の機能のみ、形式は同じ。 ・座標は論理座標に対して行われる。 ・領域色はパレット・コードまたはタイル・パターンで与える。
レベル ₃ ・IF8 ₂ と同じ。	
C180 —	MB16 ・本文の①の機能のみ、座標は絶対座標。STEPにより相対座標。
M243 —	IBM55 ・MLTと同じ。ただし、色は黒(0)、黄緑色(1)の2色である。
MZB —	
BUB ・IF8 ₂ と同じ。	PC100 ・②のバック・グラウンド・ストリングの指定がない場合は本文と同じ。
FM8 ・IF8 ₂ と同じ。	
PSP —	MSX ・形式は①のみである。 ・高解像度グラフィック・モードでは境界色は書けず、領域色を境界色とみなす。
PC88 ・本文参照。	
N52 —	
PC6 ・IF8 ₂ と同じ。 ・ワールド座標はできる。	
MLT PAINT [STEP] (80, 80),	

ドット・セット PSET

ピー・セット

形式：PSET (20, 40, 4)

省略可

機能：画面上の指定位置にドットを点灯する。

- 特徴：1 第1の引数は、座標の水平位置を示し、0～(桁数×2-1)までの整数で、第2の引数は座標の垂直位置を示し、0～(行数×4-1)までの範囲の整数である。
- 2 第3の引数は色、機能を指定するファンクション・コード(COLOR文と同じ)を指定できる。
- 3 グラフィック・モードで使用しないと、正しい動作をしないことがある。

プログラム例：ドットでサイン関数を $-\pi \sim \pi$ の領域でグラフィック表示する。

```
10 PI=3.14159:LOCATE 0,0
20 FOR IX=-PI TO PI STEP 2*PI/40
30 X=20/PI*IX+40+.5
40 Y=30-SIN(IX)*15+.5
50 PSET(X,Y)
60 NEXT IX:END
```

実行例



参照：PRESET, POINT

APL PLOT 20, 40 または、
HPlot 100, 100
・PLOTは低解像度、HPlotは高解像モードで使う。色は現在のCOLOR (HCOLOR)文で指定されたものである。座標の大きさは、39×47である。ほかは本文と同じ。

TRS₁ ・本文と同じ。ただし、引数で座標のみ指定できる。また座標の大きさは、128×48である。

CPM —

M223 —

MZK SET 40, 20
・色、機能を指定はできない。座標の大きさは、79×49である。ほかは本文と同じ。

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 ・MZKと同じ。ただし、座標の大きさは、79×31である。

IF8₂ ・本文と同じ。ただし、3番目の引数でドットの色を指定する。省略すると、現在の文字色のドットが設定される。
・座標の大きさは639×199 (含0)。

レベル₃ ・IF8₂と同じ。

C180 —

M243 —

MZB ・MZKと同じ。座標の大きさは、319×199である。

BUB PSET (50, 100), 7
・機能はIF8₂と同じ。

FM8 PSET (50, 100, 4, OR)
・機能はIF8₂と同じ。ただし、4番目の引数でAND, OR, XORの指定ができ(省略も可)現在の画面色との論理演算を行い、その結果の色を表示する。

PSP ・本文と同じ。座標が画面を越えた時は、右端または最下行に設定する。

PC88 PSET STEP (20, 40), 5
・X, Y座標とも-32768～32768まで指定できる。座標の前にSTEPを付けると相対座標も使える。最後に指定された座標(Last referenced Point)を設定できる(相対座標はこのLPに対して指定する)。

N52 —

PC6 ・本文と同じ。

MLT ・PC88と同じ。ただし、 $0 \leq X \leq 399$, $0 \leq Y \leq 639$ 。また、PRESETも同じ。だがカラーを省略した場合に異なった命令となる。
・PSET, PRESETでカラーを省略すると、前者はフォア・グラウンド色、後者はバック・グラウンド色でドットを点灯する。

HC ・本文と同じ。

FP11 PLOT (20, 40), 4, AND
・指定は座標、色、機能の順。
・DRAW文、DRAWC文を使っても同様のことができる。

SMC GPLOT (20, 40), 4, 3

ドット・セット PSET

ピー・セット

<p>・指定は LINE 文と同じ。</p> <hr/> <p>MZ35 PSET 20, 40, 4 ・機能は本文と同じ。 SET 20, 40 ・グラフィック画面を使用せず、テキスト画面で点や線を表現するときを使う。</p> <hr/> <p>X1H4 PSET (40, 40, C) ・論理座標に対して行われる。 ・引数は X, Y 座標, パレット・コードである。</p> <hr/> <p>MB16 PSET (20, 20), 4 ・4 はドットの色指定 0 ~ 7 の範囲。 ・座標の前に STEP をつけることにより相対座標。</p> <hr/> <p>IBM55 ・MB16 と同じ。ただし、色指定は 0 または 1 のみである。</p> <hr/> <p>PC100 ・PC88 と同じ。</p> <hr/> <p>MSX PSET (X, Y), M ・M は 0 ~ 15 までのカラ・コード。 ・座標の前に STEP を指定すると相対座標。</p> <hr/>			
--	--	--	--

ドット・リセット PRESET

ピー・リセット

形式：PRESET (20, 40)

機能：画面上の指定位置のドットを消去する。

特徴：1 ファンクション・コードを使わない場合は、PSET の場合と同じ。

プログラム例：グラフィック・パターンを配列 A% に読み込み、PUT @ で表示して、それを反転する。反転させる際、PRESET で表示ドットを消している。

```
100 DIM A%(10)
110 CONSOLE 0,25,0,1:COLOR 7,0,1
120 FOR I=0 TO 5:READ A%(I):NEXT I
130 PUT@ (0,0)-(7,15),A%,PSET:LOCATE 0,10
140 FOR I=0 TO 7
150 FOR J=0 TO 15
160 IF POINT(I,J)=0 THEN PSET (I,J) ELSE PRESET(I,J)
170 NEXT J,I
180 END
190 DATA 128,8704,10786,7230,5148,5140
```

実行例



参照：PSET, POINT

APL —

TRS_I ・本文と同じ。

CPM —

M223 —

MZK RESET 160, 100
・X, Y座標は、それぞれ 0～79, 0～49まで指定できる。

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 ・MZKと同じ。X, Y座標は、それぞれ 0～79, 0～31まで指定できる。

IF8₂ PRESET (30, 40, 7)
・第1の引数の水平座標と、第2の引数の垂直座標で指定された座標位置のドットを、第3の引数による指定色ドットに変える。
・第3の引数を省略すると、色を背景色に設定する。

レベル₃ ・本文と同じ。

C180 —

M243 —

MZB ・MZKと同じ。X, Y座標は、それぞれ 0～319, 0～199まで指定できる。

BUB ・指定位置のドットは背景色になる。

FM8 ・BUBと同じ。

PSP ・本文と同じ。座標が画面を越えた時は、右端または最下行に設定される。

PC88 PRESET STEP (20, 40), 5
・X, Y座標とも、-32768～32767まで指定できる。
・座標の前に STEP を付けると、相対座標も使える。ほかは本文と同じ。

N52 —

PC6 ・本文と同じ。

MLT ・該当なし。ただし、PRESET (20, 40) で同じことができる。

HC ・本文と同じ。

FP11 —

SMC —

MZ35 PRESET 20, 40
・機能は本文と同じ。
RESET 20, 40
・SET で描いた点や線を消す。

X1Hu PRESET (40, 40, C)
・論理座標に対して行われる。
・引数は PSET と同じ。

MB16 PRESET (20, 20), 4
・色指定を省略すると背景色となる。

IBM55 ・MB16と同じであるが、色指定は 0 または 1 で、省略すると 0 (黒) である。
・座標の前に STEP をつけることにより相対座標。

PC100 ・PC88と同じ。

ドット・リセット PRESET

ピー・リセット

<p>MSX ・指定位置のドットを COLOR 文で指定された背景色に する。 ・カラー・コードを指定すると、 PSETと同じ働きをする。</p>			
--	--	--	--

ドット判定 POINT

ポイント

形式：POINT (20, 40)

機能：画面上の指定位置のドットの有無を調べる。

- 特徴：1 指定位置にドット・セット（光っている）のとき真（-1），リセットのとき偽（0）を得る関数である。
 2 第1の引数は、座標の水平位置を指定し、0～(桁数*2-1)までの整数で、第2の引数は、座標の垂直位置を指定し、0～(行数*4-1)までの整数である。
 3 グラフィック処理のとき用いる。

プログラム例：グラフィック・パターンを配列A%に読み込み、PUT@で表示して、それを反転するプログラム。

```
100 DIM A%(10)
110 CONSOLE 0,25,0,1:COLOR 7,0,1
120 FOR I=0 TO 5:READ A%(I):NEXT I
130 PUT@ (0,0)-(7,15),A%,PSET:LOCATE 0,10
140 FOR I=0 TO 7
150 FOR J=0 TO 15
160 IF POINT(I,J)=0 THEN PSET (I,J) ELSE PRESET(I,J)
170 NEXT J,I
180 END
190 DATA 128,8704,10786,7230,5148,5140
```

実行例



参照：PSET, PRESET

APL —

TRS_I ・本文と同じ。

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 ・本文と同じ。ただし、指定位置にドット・セットのとき1、リセットのとき0になる。

IF8₂ ・本文と同じ。ただし、指定座標の色コードを求める関数で、水平座標は、0～639、垂直座標は0～199までである。

レベル₃ ・本文と同じ。

C180 —

M243 —

MZB ・本文と同じ。ただし、グラフィック・エリア1、2とも、ドット・リセットのとき0、1のみセットのとき1、2のみセットのとき2、双方セットのとき3の値をとる。

BUB ・IF8₂と同じ。

FM8 ・水平座標は0～639、垂直座標は0～199。

PSP ・本文と同じ。ただし、値はグラフィック・モードでは、色

コード、ファイン・グラフィック・モードのとき、OFFで0、ONで1となる。

PC88 ・本文と同じ。ただし、カラー・モードでは、指定座標位置の色コードの値をとる。
 ・指定座標が画面上にないとき-1となる。
 ・白黒モードでは本文と同じ。

N52 —

PC6 ・指定座標位置の色コードの値をとる。

MLT ・PC88と同じ。ただし、白黒モードはない。

HC ・真のときは（1）を表示する。

FP11 ・座標はユーザ座標で示す。

SMC GPOINT (20, 40)
 ・指定座標位置のカラー・コードを返す。

MZ35 ・PC3と同じ。

X1Hu POINT (20, 40)
 ・座標は論理座標。
 ・パレット・コードを与える。

MB16 ・返される値は0～7。
 ・座標の前にSTEPをつけると相対座標。

IBM55 ・PC3と同じ。ただし、座標が画面上にないときは-1。
 ・水平、垂直座標の値は、使用ディスプレイによって異なる。

PC100 ・本文と同じ。なお、ほかにPOINT (A)という形式で最終

ドット判定 POINT

ポイント

<p>参照点の座標を得るような使い方もある。</p>			
<p>MSX ・ PC6 と同じ。</p>			

指定角度, 指定の大きさの文字列を表示 SYMBOL

シンボル

形式: SYMBOL (X1, Y1), E\$, M1, M2
, M3, M4, PSET

省略可

機能: 画面上の任意の位置に文字列を指定角度, 指定サイズで表示する。

- 特徴: 1 M1 および M2 は表示する文字の横および縦の倍率を示し, 倍率×8のドット数で画面に表示される。
2 M3 はカラー・コードを表す。
3 M4 は角度コードで, 文字を回転する角度を示す。0 でノーマル, 1 で90°左回転, 2 で180°左回転, 3 で270°左回転。
4 機能として最後の引数に PSET, PRESET, AND, OR, XOR, NOT の指定ができる。

プログラム例: 文字“A”をさまざまな形で表示する。

```
10 SYMBOL (200,120), "A", 1, 1
20 SYMBOL (250,120), "A", 2, 1
30 SYMBOL (300,120), "A", 2, 2
40 SYMBOL (350,120), "A", 3, 2
50 SYMBOL (200,160), "A", 3, 2, 4, 0, PSET
60 SYMBOL (250,160), "A", 3, 2, 4, 1, PSET
70 SYMBOL (300,160), "A", 3, 2, 4, 2, PSET
80 SYMBOL (350,160), "A", 3, 2, 4, 3, PSET
```

実行例

Ready
RUN
Ready

A A A A
A [U]
A]

参照:

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 ・ 本文参照。

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC ・ 該当なし。ただし, CPLOT により指定位置に文字の表示ができる。

MZ35 GCHR X1, Y1, E\$, M4, M3, M1, M2
・ 特徴 4 以外は本文と同じ。

X1Hu CSIZE 0
・ 文字の大きさを与える。
・ 引数は,
0 または省略 ノーマル
1 垂直 2 倍, 2 水平 2 倍,
3 垂直水平 2 倍
・ PRINT # 0 で表示する。

MB16 —

IBM55 —

PC100 —

MSX —

グラフィック・カーソル座標の読み取り GCURSOR

ジー・カーソル

形式：GCURSOR (X, Y), (X1, Y1)
 , (X2, Y2), M

省略・9回まで 省略可
 の繰り返し可

機能：画面上のグラフィック・カーソルのドット座標を読み取る。

- 特徴：1 (X, Y)は最初にグラフィック・カーソルを表示する座標を指定する。
 2 (X_n, Y_n) (nは最大10)は現在表示されているグラフィック・カーソルのドット座標を読み込む変数を指定する。座標の読み込みは、(RETURN)キーが押された時行われる。
 3 グラフィック・カーソルの移動はカーソル・キーで行い、指定したn個のドット座標が読み終わるとグラフィック・カーソル表示が消える。
 4 Mはカラー・コードを示す。
 5 グラフィック・カーソルを用いることにより、ライト・ペンあるいはデジタイザ的な使い方が可能である。

プログラム例：画面上で指定した3点を読み、連結して表示する。

```
5 DEFINT X-Y
10 GCURSOR (200,100), (X1,Y1), (X2,Y2), (X3,Y3), 3
20 CONNECT (X1,Y1)-(X2,Y2)-(X3,Y3), 4
30 END
```

実行例

Ready
 RUN
 Ready



参照：

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 ・本文参照。

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 GCURSOR X, Y, X1,
 Y1, X2, Y2, M
 ・nの最大値は特に限定されていない。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

画面のセーブ GET@

形式: GET@ (0, 0) - (20, 40), B%, G

省略可

機能: 画面上の任意の範囲の文字(キャラクタ¹⁾、またはドット・グラフィック²⁾を配列にとり込む。

- 特徴: 1 画面上の任意の位置 (X1, Y1), (X2, Y2) 間の直線を対角線とする四角形の内部を、配列内に読み込む。
 2 X1, X2は画面の水平座標, Y1, Y2は垂直座標を表す。
 3 , Gを付けた時は、ドット・グラフィックの読み込みを行い、X1, X2は0から桁数*2-1まで、Y1, Y2は、行数*4-1までである。
 4 , Gを付けない時は、キャラクタの読み込みを行い、X1, X2は0から行数-1まで、Y1, Y2は行数-1までの値をとる。
 5 配列は一次元の整数型で、1キャラクタを配列の一つにとり込むので、指定した四角形内のキャラクタ数以上の大きさに指定しなければならない。
 6 配列名は、整数型に指定した方がメモリを節約でき、実行時間も速くなる。画面の絵を動かすときに使うと便利である。

用語: ¹⁾キャラクタ 数字、文字または特殊記号の実際の、あるいは符号化した表現。
²⁾ドット・グラフィックス グラフィック画面において点(ドット)の集まりで作られた文字や模様。

³⁾ユーザ座標 CRT上の原点, XY方向の倍率, XY方向の正負の向きを自由に設定できる座標。

プログラム例: 画面からパターンを読み込み、繰り返し出力する。

```
100 DIM B%(25):LOCATE 0,0
110 PRINT"  ▲ "
120 PRINT"/■■■ \"
130 PRINT"  ♥  "
140 PRINT"  / \  "
150 GET@ (0,0)-(5,3),B%
160 FOR I=1 TO 5:IX=I*3:IY=(I MOD 2)*5
170 PUT@ (IX,IY)-(IX+5,IY+3),B%
180 NEXT I:LOCATE 0,10
190 END
```

実行例



参照: PUT@

ゲット@

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ GET@ (0, 5) - (31, 15),
 A, 7, G

・本文と同じ。ただし、色情報はカラー・コードで指定できる。また、Gを付けると1ドット毎に配列内にとり込み、省略すると8ドット毎に取り込む。

レベル₃ —

C180 —

M243 —

MZB —

BUB ・IF8₂と同じ。水平座標は0~639、垂直座標は0~199まで指定できる。

FM8 ・形式は本文と同じ。、Gを付けると、後に0~8までカラー・コードを指定でき、画面のドット・カラーと一致している時点灯する。、Gを付けて後のカラー・コードを省略すると、読み込んだデータが背景色と一致しなければ点灯す

る。、Gを付けない時は本文と同じ。

PSP —

PC88 ・本文と同じ。ただし、グラフィック座標でのみ使うので、Gは付けない。また第2の座標の前にSTEPを付けると、相対座標も使える。

N52 —

PC6 ・PC88と同じ。

MLT GET (0, 0) - (20, 40),
 B1%
 ・機能は本文と同じ。

HC —

FP11 ・、Gはつけない。また、座標はユーザ座標³⁾で示す。

SMC CGET (X1, Y1) - (X2, Y2), B% (0)

・キャラクタ表示面上の画面セーブを行う。
 ・配列の添字は、どこから取り込むかを指定する。

GGET (X1, Y1) - (X2, Y2),
 B% (0)

・グラフィック画面のセーブを行う。
 AGET (X1, Y1) - (X2, Y2),
 B% (0)
 ・キャラクタ表示面上のアトリビュート情報を配列に取り込む。

MZ35 GENTER 0, 0, 20, 40,
 A@ (*)

・グラフィック画面の指定領域内に表示されているグラフィック・パターンを入力する。
 ・特徴の1, 2は本文と同じ。
 ・配列変数は三つある。

画面のセーブ GET@

ゲット@

<p>X1Hu GET@(0, 0)-(20, 20), B%, 7 ・グラフィック画面を取り込む。 ・第4 引数はバレット・コードを与える。省略するとテキスト画面。</p>			
<p>MB16 ・PC88と同じ。</p>			
<p>IBM55 ・MLT と同じ。ただし、 G はつけない。</p>			
<p>PC100 ・PC88と同じ。</p>			
<p>MSX —</p>			

画面のセーブ GET@A

ゲット@エー

形式：GET@A (X1, Y1) - (X2, Y2), B%

機能：画面上の任意の範囲の表示，機能をそのまま配列にとり込む。

- 特徴：
- 1 キャラクタとドット・グラフィックを同時に配列内にとり込むことができる。
 - 2 水平座標 X1, X2 は，0 から桁数-1 まで，垂直座標 Y1, Y2 は，0 から行数-1 までの値である。
 - 3 配列の大きさは GET@ の 2 倍必要。
その他は，GET@ と同様。
 - 4 繰り返しの図を描くとき便利である。

プログラム例：グラフィック・パターンとキャラクタ・パターンを画面に出力して，配列 B% に GET@A で読み込んで，PUT@A で繰り返し出力する。

```
100 DIM A%(10),B%(50):COLOR 7,0,1
110 FOR I=0 TO 5:READ A%(I):NEXT I
120 LOCATE 0,5
130 PRINT"  ▲  "
140 PRINT"/■■\ "
150 PRINT"  ♥  "
160 PRINT"  /\  "
170 PUT@A(2,4)-(9,19),A%,PSET
180 GET@A(0,1)-(4,8),B%
190 FOR I=0 TO 2
200 PUT@A(I*5+5,1)-(I*5+9,8),B%
210 NEXT I:END
220 DATA 128,8704,10786,7230,5148,5140
```

実行例



参照：PUT@A, GET@

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 ・本文と同じ。

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu ・GET@A はないが GET@ でバレット・コードを省略すれば，テキスト画面のキャラクタと属性を取り込める。

MB16 —

IBM55 —

PC100 —

MSX —

スプライト・パターンの登録 SPRITE\$

スプライト\$

形式：SPRITE\$(M) = “文字列”

機能：スプライト・パターンを指定する。

特徴：1 SCREEN 文のスプライト・サイズの値により、パターン番号
文字列の長さが決まる。

	パターン番号	文字列の長さ
0, 1	256未満	8バイト
2, 3	64未満	32バイト

2 模様はドットのパターンと、文字のコードのビット列を対応させて表す。

(例)

●●●●●●●●→16進数の FF

○○●●●○○●→16進数の 99

●●●●●●●●→16進数の FF

これをパターン1として指定するには、SPRITE\$(1)=CHR\$(&HFF) + CHR\$(&H99) + CHR\$(&HFF) + ...とする。

プログラム例：ドット・パターンを SPRITE\$ に設定し、画面に表示する (スプライト・サイズは1)。

```

100 SCREEN 3.1
110 SPRITE$(1)=CHR$(24)+CHR$(36)+CHR$(36)
      +CHR$(231)+CHR$(255)+CHR$(195)
      +CHR$(195)+CHR$(195)
120 SPRITE$(2)=STRING$(8,CHR$(255))
130 PUT SPRITE 1.(100.100).15.1
140 PUT SPRITE 2.(110.110).8.2
150 PUT SPRITE 3.(120.120).11.1
160 GOTO 160

```

参照：PUT SPRITE

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX ・本文参照。

配列画面出力 PUT @

プット @

形式: PUT@ (0, 0) - (20, 40), B%, PSET

機能: キャラクタまたはドット・グラフィックを画面の指定位置に表示する。

- 特徴: 1 画面上の任意の位置 (X1, Y1) - (X2, Y2) 間の直線を対角線とする四角形の内部に、GET@ で配列にセーブしたものを表示させる。
- 2 キャラクタとしてセーブされたものを出力する際、座標の水平位置 X1, X2 は 0 から桁数 - 1 まで、垂直位置 Y1, Y2 は 0 から行数 - 1 までの値をとる。
- 3 2 の際、第 3 の指定で色や機能を指定するファンクション・コードを付けることができる(省略も可)。
- 4 ドット・グラフィックとしてセーブされたものを出力する時、X1, X2 は 0 から桁数 * 2 - 1 まで、Y1, Y2 は 0 から行数 * 4 - 1 までである。
- 5 4 の際、第 3 の指定で、PSET, PRESET, OR, AND, NOT, XOR のいずれかを付けなければならない。PSET のとき配列データを画面に表示し、PRESET のとき消去する。OR, AND, XOR のとき配列データと表示との間で論理演算を行う。NOT は配列データの論理否定を行う。
- 6 通常は、GET@ と対にして使う。

プログラム例: グラフィック・パターンを配列に読み込んで、PUT @ で繰り返し出力する。

```

10 DIM A%(4)
20 FOR I=0 TO 4
30 READ A%(I)
40 NEXT I
50 FOR J=0 TO 2
60 PUT@ (16*J, 0) - (16*J+15, 3), A%, PSET
70 NEXT J
80 DATA 64, 119, 81, 81, 247

```

実行例

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ PUT@ (0, 0) - (20, 40), B%, 7, PSET

- ・機能は本文と同じ。ただし、3 番目の引数は色コードで、GET@ と共に使うときは、GET@ で取り込んだ色コードを指定しなければならない。@ は省略できる。

レベル₃ —

C180 —

M243 —

MZB —

BUB ・IF8₂と同じ。ただし、色コードを省略すると、現在の色に設定される。

FM8 PUT@ (0, 0) - (100, 200), B%, 4

- ・機能は本文と同じ。ただし、PSET などは付けない。最後の指定で文字の色を指定する。省略すると、現在の色に設定される。

PSP —

PC88 PUT@ (120, 700), B%, PSET

- ・指定座標を始点として配列用データを表示する。配列名の後に配列の表示開始番号を指定できる。省略すると配列の最初から始める。
- ・4 番目の引数で、白黒モードで取り込まれた白色をカラー・モードで使う際の色コードを指定する。また、5 番目の引数で同様に、黒をカラー・モードに変える際の色コードを指定できる。
- ・配列名を特に KANJI (漢字コード) と指定すると、コードで指定された文字を表示する。@ は省略できる。

N52 —

PC6 PUT@ (X, Y), B%, PSET

- ・機能は本文と同じ。

MLT PUT (120, 70), B%, PSET

- ・通常 GET と対で用いる。
- ・位置は長方形の左上の点。
- ・機能を省略すると XOR になる。
- ・PSET (省略可) は配列中の情報をそのまま表示し、PRESET は反転して表示する。
- ・特徴 5 において、NOT はない。また、論理演算の結果のカラー・コードでドットを表示する。

HC —

FP11 ・座標はユーザ座標で示す。また、機能指定は AND, OR, XOR のみ。

SMC CPUT (X1, Y1) - (X2, Y2), B% (0)

参照: GET@, PUT SPRITE, SPRITE\$

配列画面出力 PUT@

<ul style="list-style-type: none"> ・キャラクタ表示面上の指定領域に、配列に取り込まれたキャラクタを表示する。 <p>GPOT (X1, Y1) - (X2, Y2), B% (0), 0, 4</p> <ul style="list-style-type: none"> ・グラフィック画面上の指定領域に、配列に取り込まれたドット・グラフィックスを表示する。 ・第3の指定は論理演算コードで、第4の指定は透明色コードである。 <p>APUT (X1, Y1) - (X2, Y2), B% (0)</p> <ul style="list-style-type: none"> ・キャラクタ表示面上の指定領域のアトリビュートを、配列に取り込まれたアトリビュート情報によって変更する。 <p>KPRINT (座標), 文字幅指定コード, 漢字コード</p> <ul style="list-style-type: none"> ・グラフィック画面上に漢字を表示する。 	<ul style="list-style-type: none"> ・機能を省略すると XOR, <hr/> <p>IBM55 ・MLT と同じ、ただし、グラフィック・モードのみである。</p> <hr/> <p>PC100 ・PC88 の機能にキャラクタ表示機能を追加したものである。</p> <hr/> <p>MSX ・該当なし、ただし、PUT └─SPRITE 文により、SPRITE \$ 変数に保持されたドット・グラフィックを表示できる。PUT└ SPRITE, SPRITE \$ 参照。</p> <hr/>		
<p>MZ35 KDISP A\$ A\$ = CHR\$ (&4C24, &4D68)</p> <ul style="list-style-type: none"> ・CRT ディスプレイに漢字を出力する。 ・漢字だけでなく、漢字とキャラクタが混在した場合でも出力できる。 <p>KPRINT A\$ A\$ = CHR\$ (&4C24, &4D68)</p> <ul style="list-style-type: none"> ・プリンタに漢字を出力する。 <p>GDISP 0, 0, 20, 40, B@</p> <ul style="list-style-type: none"> ・CRT にグラフィック・パターンを表示する。 ・配列変数に代入しておいたグラフィックスの情報を、CRT に表示する。 ・特徴1 だけある。 			
<p>X1Hu PUT@ (0, 0) - (10, 10), PSET, 7</p> <ul style="list-style-type: none"> ・最後の引数でパレット・コードを指定する。 <hr/> <p>MB16 PUT(40, 50), A, PSET</p> <ul style="list-style-type: none"> ・座標は左上すみの絶対座標。 			

配列画面出力 PUT@A

プット@エー

形式：PUT@A (X1, Y1) - (X2, Y2), B%

機能：キャラクタとドット・グラフィックを、同時に画面の指定位置に出力する。

- 特徴：1 座標の水平位置 X1, X2 は、0 から指数-1 までの値、座標の垂直位置 Y1, Y2 は、0 から行数-1 までの値をとる。
2 GET@A で配列にとり込まれたものを、色や機能も含めて表示する。
3 繰り返しの図を描くとき便利である。

プログラム例：グラフィック・パターンとキャラクタ・パターンを画面に出力して、配列 B% に GET@A で読み込んで、PUT@A で繰り返し出力する。

```
100 DIM A%(10), B%(50):COLOR 7,0,1
110 FOR I=0 TO 5:READ A%(I):NEXT I
120 LOCATE 0,5
130 PRINT " "
140 PRINT " "
150 PRINT " "
160 PRINT " "
170 PUT@A(2,4)-(9,19),A%,PSET
180 GET@A(0,1)-(4,8),B%
190 FOR I=0 TO 2
200 PUT@A(I*5+5,1)-(I*5+9,8),B%
210 NEXT I:END
220 DATA 128,8704,10786,7230,5148,5140
```

実行例



参照：GET@A, PUT@

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 ・本文と同じ。

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu ・PUT@でできる。

MB16 —

IBM55 —

PC100 —

MSX —

変数画面出力 PUT SPRITE

ブット・スプライト

形式: PUT SPRITE M1, STEP (X, Y), M2, M3

省略可

省略可

機能: 画面上の指定位置に SPRITE\$ に指定されたドット・パターンを表示する。

- 特徴:
- 1 M1 により, スプライト・パターンを表示する面を 0~31 の数値で指定する。
 - 2 スプライト面 0 が最も優先順位が高く, スプライト面 31 が最も優先順位が低い。
 - 3 二つ以上のスプライトが重なった場合, 優先順位の低いスプライトの重なった部分が消える。
 - 4 STEP を省略すると, 絶対座標となる。
 - 5 M2 によりカラーを 0~15 の数値で指定する。省略した場合はフォア・グラウンド・カラーとなる。
 - 6 M3 により, SPRITE\$ 文で定義したパターン番号を指定する。
 - 7 ON SPRITE GOSUB 文により, スプライト・パターンの重なりによる割り込みが可能である。
 - 8 SPRITE ON/OFF/STOP により, 割り込みの実行, 禁止, 一時停止を行う。

プログラム例: SPRITE\$ 参照。

参照: SPRITE\$

APL —

TRS —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS —

PC3 —

IF8₂ —レベル₃ —

CI80 —

M243 —

MZB PATTERN -16, A\$
 ・第1引数で段数, 第2引数でドット・パターンを指定して画面に描く。
 ・スプライト機能はない。
 POSITION 0, 0
 ・PATTERN 文を実行する画面上の位置を決める。

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ・ MZB と同じ。

XIH_u PATTERN (-16,
 KANJI\$ (&H7F0))
 ・第1引数で段数, 第2引数で文字
 パターンを指定して画面に描く。
 POSITION 0, 0
 ・PATTERN 文を実行させる画面
 上の位置を決める。
 KANJI\$ (&H7F0)
 ・引数で与えられた漢字パターンを
 得る。

MB16 —

IBM55 —

PC100 —

MSX ・ 本文参照。

ブザー BEEP

ビーブ

形式: BEEP 1

□
省略可

機能: ブザー音を鳴らす。

- 特徴: 1 引数を1と指定するとブザーを鳴らし、0と指定することでブザーを止めることができる。
- 2 省略すると、一定時間(約450ms)ブザーを鳴らし、止める。
- 3 BEEP1を実行すると、BEEP0を実行するまでブザーは鳴り続く。(STOP) キーはきかない。

プログラム例: 一定の時間間隔を置いてブザーを5回鳴らし、それぞれ鳴り始めに BUZZER ON, 鳴り終わりに BUZZER OFF と出力する。

```
10 FOR I=1 TO 5
20 BEEP1:PRINT"BUZZER ON ";
30 FOR J=1 TO 10:PRINT"-";NEXT J
40 BEEP0:PRINT" BUZZER OFF"
50 FOR K=1 TO 500:NEXT K
60 NEXT I:END
```

実行例

```
BUZZER ON ----- BUZZER OFF
BUZZER ON ----- BUZZER OFF
BUZZER ON ----- BUZZER OFF
BUZZER ON ----- BUZZER OFF
BUZZER ON ----- BUZZER OFF
```

参照: MUSIC

APL —

TRS: —

CPM —

M223 PRINT CHR\$(7)

MZK MUSIC "DE#FGA"
・TEMPO文で指定した速さで、引用符(")で囲まれた文字列で指定した音程と音長で演奏を行う。

CBM —

PC8 ・本文参照。
・PRINT CHR\$(7)でBEEPと同様のことができる。

TRS: —

PC3 MUSIC T1, 1, C
・引数により音の速度、音符、高さ、半音、休符の指定などで演奏を行う。

IF82 BEEP
・本文と同じ。ただし、引数は使えない。

レベル3 ・本文と同じ。ただし、引数は0~255までの値をとれるが意味はない。省略もできる。

C180 BUZZER
・IF82と同じ。約50ms鳴った後止まる。

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・IF82と同じ。

FM8 ・本文と同じ。

PSP ・IF82と同じ。1秒鳴った後止まる。

PC88 ・本文と同じ。

N52 ・IF82と同じ0.2秒鳴った後止まる。

PC6 —

MLT ・M223と同じ。

HC SOUND 10, 5
・第1の引数は音程0~56。
・第2の引数は長さで、長さ×0.1秒間鳴る。

FP11 BEEP
・ブザーを鳴らす。
BEEP ON
・キー入力音を発生させる。
BEEP OFF
・キー入力音を消す。

SMC ・IF82と同じ。

MZ35 —

X1Hu ・本文と同じ。

MB16 ・IF82と同じ。また、PRINT CHR\$(7)でも同様。

IBM55 ・MB16と同じ。なお、1/4秒鳴った後止まる。

PC100 ・1, 0の指定はなく、特徴2の機能をもつだけである。

MSX ・IF82と同じ。

音楽演奏 MUSIC

ミュージック

形式：MUSIC E\$
 ↑
 必要

機能：コーテーション間の音を自動演奏する。

- 特徴：1 TEMPO と組み合わせる。
 2 音名、休符、音長、オクターブを E\$ で指定する。
 3 音名は CDEFGAB と半音を表す # を前につけて指定する。低音域は□、高音域は+ を前につける。
 4 休符は R、長さの指定は音符と同じ。
 5 音長は32分音(休)符を0、16分音(休)符を1、…、全音(休)符を9とし、それらを音名の後につけて指定する。前の音長が続くときは省略できる。最初から指定がないと5。
 (例) E\$ = "C 1 □ C □ E □ # C"

プログラム例：TEMPO と組み合わせドレミファソラシと演奏。

```
10 TEMPO 4
20 MUSIC "CDEFGAB"
30 END
```

参照：TEMPO

APL —

TRS_I —

CPM —

M223 —

MZK ・本文参照。

CBM —

PC8 —

TRS_{II} —

PC3 ・本文と同じ。ただし、
TEMPO は使用しない。

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB ・本文と同じ。ただし、低
音域は-、高音域は+ を前につけ
る。

BUB —

FM8 —

PSP —

PC88 —

N52 PLAY E\$
 ・E\$ でオクターブ、音名、半音、
長さ、アクセントなどのデータを
指定する。

PC6 ・N52 と同じ。

・これに関連して、効果音を発生さ
せる。SOUND 命令がある。

MLT —

HC —

FP11 —

SMC ・N52 と同じ。ただし、
E\$ でテンポ、音の長さ、オク
ターブ、音程、休符を指定できる。

MZ35 ・テンポも E\$ 中で指定で
きる。

X1Hu ・本文と同じ。ただし、オ
クターブ指定は 01~08、音量は
V0~V15 で指定する。
・PLAY 文で MUSIC 文と TEMPO
文の両方の機能をもつ。
・PC6 の機能もある。

MB16 ・N52 と同じ。

IBM55 ・N52 と同じ。

PC100 ・N52 と同じ。

MSX ・SMC と同じ。
 ・PLAY (M) 関数により演奏状態
を知ることができる。
 ・SOUND 命令により、PSG (プロ
グラマブル・サウンド・ジェネレ
ータ) のレジスタに直接データを
書き込み演奏ができる。

テンポの指定 TEMPO

テンポ

形式：TEMPO A

↑
必要 単精度変数または整数

機能：音楽演奏のテンポ（速さ）を指定する。

- 特徴：1 MUSIC と組み合わせて使う。
2 引数のAは、1 から7 までの数を指定する。
3 速さはおよそ次のとおり(実測)。
" 1 " ... J =130 (毎分)
" 2 " ... J =150
" 3 " ... J =176
" 4 " ... J =230
" 5 " ... J =300
" 6 " ... J =500
" 7 " ... J =1000

プログラム例：MUSIC と組み合わせドレミファソラシと演奏。

```
10 TEMPO 4
20 MUSIC "CDEFGAB"
30 END
```

参照：MUSIC, PLAY

APL —

TRS₁ —

CPM —

M223 —

MZK ・ 本文参照。

CBM —

PC8 —

TRS₁₁ —

PC3 ・ 速度は本文の $\frac{1}{2}$ ～ $\frac{1}{4}$ ぐらい。
い。

IF8₂ —

レベル₃ —

C180 —

M243 —

MZB ・ 本文と同じ。

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 ・ PLAY 参照。

MLT —

HC —

FP11 —

SMC ・ PLAY 参照。

MZ35 —

X1Hu ・ Aは30～7500。
・ 数値は1分あたり4分音符の拍数。

MB16 —

IBM55 ・ PLAY 参照。

PC100 —

MSX —

オープン

入出力ポートのオープン文 OPEN_{入出力}

形式：OPEN "POUT" FOR OUTPUT AS
 FILE 1 MODE 3
 省略可

機能：入出力ポートをオープンし、ファイル番号と結びつける。

- 特徴：1 この命令でオープンした入出力ポートは、以後ファイルと同じ概念で扱える。
 2 モード番号 (MODE の後の数字) は通常 3 を指定する。
 3 入出力ポートを使用する時には、必ずこの命令を実行しなければならない。
 4 ファイル番号は、1～8。
 5 RS-232C ポートのオープンにも用いる。

プログラム例：最初にプリンタの接続されたパラレル・ポートにチャネル番号 1 を割りふり、そこへ出力してから CLOSE する。

```
10 OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20 FOR I = 1 TO 5
30 PRINT #1, "SQR(" ; I ; ")=" ; SQR(I)
40 NEXT I
50 CLOSE 1
60 END
```

実行例

```
SQR( 1 ) = 1
SQR( 2 ) = 1.41421
SQR( 3 ) = 1.73205
SQR( 4 ) = 2
SQR( 5 ) = 2.23607
```

参照：CLOSE

APL —

TRS_I —

CPM —

M223 ・本文参照。

MZK —

CBM OPEN 1, 4

・プリンタ、ディスプレイ、カセット、IEEE バス、キーボード、ディスクなどのオープンができる。

PC8 —

TRS_{II} —

PC3 —

IF8₂ OPEN "LPT2:" AS #3

・プリンタ、RS-232-C ポート、カセット、ROM カートリッジ、ディスクなどのオープンができる。

レベル₃ OPEN "O", #1, "COM1:"

・この命令によりデータ長、パリティ、ストップ・ビットなどの指定もできる。

C180 OPEN #1: \$LP (40)

・キーボード、CRT、プリンタ、ディスク、通信回線などのオープンができる。
 ・通信回線用の関数として REQ, ST%, TID% がある。

M243 ・本文と同じ。

MZB —

BUB —

FM8 ・レベル₃と同じ。

PSP OPEN # "CM1",
 "14N03200", 1
 ・プリンタ、RS-232 C ポートのオープンができる。

PC88 OPEN "CAS1:TEST"
 FOR INPUT AS #1

・カセット、キーボード、プリンタ、ディスプレイ、RS-232 C ポートなどのオープンができる。

N52 —

PC6 —

MLT —

HC ・レベル₃と同じ。

FP11 OPEN "COM0:POUT"
 FOR OUTPUT AS #1

・プリンタ、カセット、キーボード、ディスク、ROM または RAM パック、通信回線などのオープンができる。
 ・ファイル番号は 1～15。

SMC OPEN/I, OPEN/O,
 OPEN/R, OPEN/S

・キーボード、ROM パック、プリンタ、ディスプレイ、カセット、RS-232C ポートなどのオープンができる。

MZ35 ・該当なし。ただし、CHANNEL は RS-232C の使用チャネルの設定および、データの入出力型式を設定する。

X1Hu —

MB16 OPEN "COM:1200, E,
 7, 1" AS #1

・プリンタ、キーボード、ディスク、RS-232C などオープンできる。

入出力ポートのオープン文 OPEN 入出力

オープン

<p>・この命令によって、ボーレート、データ長、パリティ、ストップ・ビットなど指定できる。</p> <hr/> <p>IBM55 OPEN "COM1:" FOR OUTPUT AS #1</p> <p>・プリンタ、キーボード、ディスプレイ、通信回線などのオープンができる。</p> <p>・OPEN ファイルのような使い方も可。</p> <hr/> <p>PC100 ・OPEN "COMn:~" により RS-232C ポートのオープンができる。</p> <hr/> <p>MSX OPEN "デバイス名:ファイル名" FOR モード AS 番号</p> <p>・カセット、プリンタ、ディスプレイなどのオープンができる。</p> <hr/>			
---	--	--	--

通信回線からの割り込み許可 COM (n) ON

コム・オン

<p>形式：COM (0) ON</p> <p style="text-align: center;">└──┘ ポート番号</p> <p>機能：コミュニケーション・ポート¹⁾の入力割り込みを許可する。</p> <p>特徴：1 ポート番号は、0～4の整数表記で指定する。 2 COM (0) ON で入力を許可する。この文を実行後は、入力があると、ON COM (0) GOSUB で指定された処理ルーチンへ制御が移る。処理ルーチンの最後に使うと（下のプログラム例のように）、通信が混んだとき割り込みが多重に入りスタック²⁾が深くなってしまうことがあるので注意。 3 COM (0) OFF で入力を禁止することができる。 4 COM (0) STOP で入力の受け付けを一時停止する。実行後は、入力があっても処理プログラムに制御は移らないが、入力があったことは記憶されている。したがって後に COM (n) ON を行った場合は、処理プログラムに制御が終る。 5 プログラム終了時には、COM OFF を実行する。 6 ON COM (n) GOSUB を使うときに必要である。</p> <p>用語：¹⁾コミュニケーション・ポート コンピュータ間、または、周辺機器との通信に利用するためのハードウェア。 ²⁾スタック プログラム実行中の制御情報などを格納しておくところ、または、その情報を格納すること。</p>	<p>APL —</p> <p>TRS_I —</p> <p>CPM —</p> <p>M223 —</p> <p>MZK —</p> <p>CBM —</p> <p>PC8 —</p> <p>TRS_{II} —</p> <p>PC3 —</p> <p>IF8₂ ・本文と同じ。ただし、ポート番号は1～4である。</p> <p>レベル₃ ・本文参照。</p> <p>C180 —</p> <p>M243 —</p> <p>MZB —</p> <p>BUB —</p> <p>FM8 ・本文と同じ。</p> <p>PSP —</p> <p>PC88 COM ON COM OFF COM STOP ・上記のすべての命令は三つのポートについて同時に有効である。</p> <p>N52 —</p> <p>PC6 —</p>	<p>MLT —</p> <p>HC —</p> <p>FP11 ・PC88と同じ。ただし、ポートは0～7について有効。</p> <p>SMC —</p> <p>MZ35 —</p> <p>X1Hu —</p> <p>MB16 ・ポート番号は1～4である。</p> <p>IBM55 ・ポート番号は1または2。</p> <p>PC100 ・ポート番号は1のみ。 ・プログラム終了時にはCOM_{OFF}のほかCOM_{STOP}でもよい。 ・その他はPC88と同じ。</p> <p>MSX —</p>
<p>プログラム例：大型計算機の端末として使用する(実行例はON COM (n) GOSUB を参照)。</p> <pre> 10 OPEN "Q",#1,"COM0:(S7E1)" 20 OPEN "I",#2,"COM0:(S7E1)" 30 ON COM(0) GOSUB B0 40 COM(0) ON 50 A\$=INKEY\$:IF A\$="" THEN GOTO 50 60 PRINT A\$;:PRINT #1,A\$; 70 GOTO 50 80 COM(0) STOP 90 B\$=INPUT\$(LOF(2),#2) 100 IF LEFT\$(B\$,1)=CHR\$(%H7F) THEN GOTO 120 110 PRINT B\$; 120 COM(0) ON 130 RETURN </pre> <p>参照：ON COM (n) GOSUB</p>		

通信回線の入力時の分岐 ON COM (n) GOSUB

オン・コム・ゴースブ

<p>形式：ON COM (0) GOSUB 70</p> <p style="text-align: center;"> ポート番号 行番号 </p> <p>機能：コミュニケーション・ポートに入力割り込み¹⁾があった時、プログラムの制御を移す番号を指定する。</p> <p>特徴：1 処理プログラムの実行後は、RETURN 文によって、メイン・プログラムの割り込みがかかった場所にプログラムの制御がもどる。RETURN 文の後に、行番号を指定したときは、その行から再開する。</p> <p>2 処理プログラムに分岐させるには、割り込みが発生した時、COM (0) ON の状態でなければならない。</p> <p>3 回線を使用するプログラムにおいて、常に入力の有無を確認する必要がないので便利である。</p> <p>用語：¹⁾割り込み 実行中のプログラムを故意に一時停止させ、その時にほかの処理をさせること。それが終了後、一時停止したプログラムが再開される。</p>	<p>APL —</p> <p>TRS_I —</p> <p>CPM —</p> <p>M223 —</p> <p>MZK —</p> <p>CBM —</p> <p>PC8 —</p> <p>TRS_{II} —</p> <p>PC3 —</p> <p>IF8₂ ・本文と同じ。ただし、ポート番号は1～4である。</p> <p>レベル₃ ・本文参照。</p>	<p>HC —</p> <p>FP11 ON COM GOSUB 70, 170, 270 ・行番号に、順にCOM0～COM7が 対応して分岐する。</p> <p>SMC —</p> <p>MZ35 —</p> <p>X1Hu —</p> <p>MB16 ・本文と同じ。</p> <p>IBM55 ・ポート番号は1または2。</p> <p>PC100 ・ポート番号は1のみ。 ・文番号の代わりにラベルも使用可。</p> <p>MSX —</p>
<p>実行例：大型計算機の端末として使用する(プログラムはCOM (n) ONを参照)。</p> <p> <pre> **COST: 0.09 TO DATE: 3917.67= 47% **ON AT 20.417 - OFF AT 20.425 ON 01/19/82 CP DISCONNECTS </pre> </p>	<p>C180 —</p> <p>M243 —</p> <p>MZB —</p> <p>BUB —</p> <p>FM8 ・本文と同じ。</p> <p>PSP —</p> <p>PC88 ON COM GOSUB 70 ・文番号の代わりにラベルが使用で きる。</p> <p>N52 —</p> <p>PC6 —</p> <p>MLT —</p>	
<p>参照：COM (n) ON, ON KEY GOSUB</p>		

端末機として使用 TERM

ターミナル

形式: TERM a, 1, 0, 1

機能: システムを端末機として使用する。

- 特徴: 1 外部の機器と RS-232C¹⁾を通して、通信できる。
 2 引数の説明

引数(左から)	種類	
1 番目	コード	a...ASCII コード, j...JIS コード
2 番目	パリティ ²⁾	0...NON, 1...ODD, 2...EVEN
3 番目	ボーレート ³⁾	0...基準周波数×64, 1...×16
4 番目	フィード・スイッチ	0...OFF, 1...ON ON の場合、キャリッジ・リターン・コードを受信すると自動的に改行する。

- 3 システムを端末機として使用したいときに使うと便利である。

用語:

¹⁾RS-232C 本来データ通信において、モデムと各種種類のインターフェースを定めた規格の名称。パソコンの場合、汎用インターフェースの性格がでている。

²⁾パリティ 2進コードにおいて、“1”の数を奇数もしくは、偶数にするように余分のビット(パリティ・ビット)を付加し、2進コードの誤りの有無を検出すること。

³⁾ボーレート(ボー) ボーとは、信号の伝送速度の単位で、1ボーは2進信号においては1ビット/1秒である。ボーレートは、これを単位とした大きさである。

⁴⁾全二重方式 データ通信において、送受信いずれの方向も伝送可能な方式。☐半二重方式。キーボードから入力した文字は画面に出ない(ホストからエコー・バック⁷⁾してくるのを出力する)。

⁵⁾半二重方式 データ通信において、送受信どちらの方向も伝送可能だが、交互に一方のみだけに使用するように構成された方式。

⁶⁾リモート BASIC プロトコル RS-232C 通信回線を用いて、ターミナル・モードになっている本体の BASIC プログラムを実行させたりすること。

⁷⁾エコー・バック 回線から情報を受信したときまたそれを確認のため送り返すこと。

実行例: 大型計算機 ACOS-600 の TSS 端末機として使用する。

```
term j,2,1,0
```

```

          TRMIAL ID - 99
PROGRAM NAME - TSS
ACOS 77 TSS ON 01/12/82 AT 17.039 CHANNEL 1111

USER ID -kiso
PASSWORD--
TXOGMEMWEPVR
565 BLOCKS FILE SPACE AVAILABLE

SYSTEM ?
```

参照: OPEN #

APL —

TRS_I —

CPM —

M223 ・ OPEN # 文で回線をオープンし、そして入出力のソフトウェアを作れば可能である。
 ・ 専用の T-BASIC が用意されている。

MZK —

CBM —

PC8 ・ 本文参照。

TRS_{II} —

PC3 —

IF8₂ ・ OPEN 文で回線をオープンし、そして入出力のソフトウェアを作れば可能である。

レベル₃ TERM “S8N2H”

・ S はクロック指定で、S (Slow クロック) または F (Fast クロック)、省略時は S である。

・ 8 はビット数の指定で、7 (7 ビット) または 8 (8 ビット) である。

・ N はパリティの指定で、N (パリティなし) または O (奇) または E (偶)、省略時は N である。

・ 2 はストップ・ビットの指定で、1 (1 ストップ・ビット) または 2 (2 ストップ・ビット) である。

・ H は通信モードの指定で、F (全二重方式) または H (半二重方式) である。

C180 ・ IF8₂ と同じ。

M243 ・ M223 と同じ。

MZB —

BUB —

FM8 TERM “S8N2HA”
 ・ 機能はレベル₃と同じ。ただし、A はオート LF の指定で、A は行い N は行わない、PC8 のフィード・スイッチと同じ。

PSP TERM 1, N, 3, 0, H
 ・ 1 は転送レートの指定で、1 ~ 5 である。

・ N はパリティの指定で、N (パリティなし) または O (奇) または E (偶) である。

・ 3 はストップ・ビットの指定で、0 (無効) または 1 (1 ビット) または 2 (1.5 ビット) または 3 (2 ビット) である。

・ 0 はキャラクタ長の指定で、0 (7 ビット) または 1 (8 ビット) である。

・ H は通信モードの指定で、F (全二重方式) または H (半二重方式) である。

PC88 TERM “COM: E71X”, H, 800

・ COM: はチャネル番号の指定で、COM1 ~ 3 である。

・ E はパリティの指定で、N または O (奇) または E (偶) である。

・ 上で 7 は 1 文字のビット数の指定で、7 または 8 ビットである。

・ 1 はストップ・ビットの指定で、1 (1 ビット) または 2 (1.5 ビット) または 3 (2 ビット) である。

・ X は XON スイッチの指定で、X は XON/XOFF による通信制御を行い、省略時は行わない。

・ H は通信モードの指定で、F (全二重方式⁴⁾) または H (半二重方式⁵⁾)

端末機として使用 TERM

ターミナル

<p>である。 ・800は、リモート BASIC プロトコル⁶⁾を利用する際に用いられる変数領域の大きさを指定する。</p>	<p>・このほか、RS-232C 専用命令として、CHANNEL, SEND, RCV, SENREV, OPCHNL, POLLING, RLIST がある。</p>		
<p>N52 —</p>	<p>X1Hu —</p>		
<p>PC6 —</p>	<p>MB16 —</p>		
<p>MLT —</p>	<p>IBM55 ・専用のプログラムがある。</p>		
<p>HC OPEN "COM0 :" ・OPEN 文で RS-232C ポートを開き、入出力命令により可能。</p>	<p>PC100 ・専用の TELCOM がある。 MSX —</p>		
<p>FP11 TERM "COM0 : (F8N2H)" ・COM0~COM7 を使用。 ・F はボーレイトの指定で、F (DIPSW の値) または S (DIPSW の 1/4 の値) である。 ・8 はビット数の指定で、7 または 8 である。 ・N はパリティの指定で、N (パリティなし) または E (偶) または O (奇) である。 ・2 はストップ・ビットの指定で、0 (1 ビット) または 1 (1.5 ビット) または 2 (2 ビット) である。 ・H は通信モードの指定で、F (全二重方式⁴⁾) または H (半二重方式⁵⁾)、 ・コロン (:) 以後を省略すると、F8N2H に設定される。</p>			
<p>SMC —</p>			
<p>MZ35 TERM 1200, "7E1", 1 ・本機を端末機化し、データの入出力形式を、ボーレイトは 1200、データ形式は 7 単位コード、偶数パリティ・ストップ・ビット 1 に設定し、CR コードに改行機能をもたせる。</p>			

タイマ割り込み許可, 禁止 TIME ON, TIME OFF, TIME STOP

タイム・オン, タイム・オフ, タイム・ストップ

形式: ① TIME ON ③ TIME STOP
 ② TIME OFF

機能: タイマ割り込みの状態を設定する。

- 特徴: 1 TIME ON のとき, TIME 文で指定した割り込み時に, ON TIME GOSUB 文で指定した処理ルーチンの開始行に分岐する。
 2 TIME OFF のとき, タイマ割り込みを禁止する。
 3 TIME STOP のときタイマ割り込みを停止する。このとき, 割り込みは受け付けられ, TIME ON が実行された後で分岐する。
 4 プログラム終了時には, TIME OFF を実行すべきである。
 5 インターバル割り込みに関しては INTERVAL ON, INTERVAL OFF, INTERVAL STOP で行う。
 6 インターバル割り込み間隔定義は, INTERVAL M1 (M1 は秒単位) で行う。

プログラム例: 現在の時刻と終了時刻を入力し, 時間を出力しながら
 終了時間に TIME OUT と出力してブザーを鳴らす。

```

10 CLS:ON TIME GOSUB 110
20 INPUT "PRESENT TIME";A$
30 INPUT "LAST TIME";B$
40 TIME$=A$
50 TIME B$
60 TIME ON
70 LOCATE 0,3:PRINT TIME$
80 GOTO 70
90 TIME OFF
100 END
110 LOCATE 0,3:PRINT TIME$
120 PRINT "TIME OUT"
130 BEEP1
140 FOR I=0 TO 200:NEXT
150 BEEP0
160 RETURN 90
  
```

実行例

```

PRESENT TIME? "15:00:00"
LAST TIME? "15:01:00"
  
```

```

15:01:00
TIME OUT
  
```

参照: TIME\$, ON TIME GOSUB

APL —	FP11 —
TRS ₁ —	SMC —
CPM —	MZ35 ON TIME OFF TIME ・機能は本文と同じ。
M223 —	X1Hu —
MZK —	MB16 —
CBM —	IBM55 —
PC8 —	PC100 ・スベルが TIME でなく TIMER である場合は本文と同じ。
TRS ₁₁ —	MSX ・INTERVAL ON/OFF/ STOP が, インターバル・タイマ 割り込みに対して使用できる。
PC3 —	
IF8 ₂ —	
レベル ₃ —	
C180 —	
M243 —	
MZB —	
BUB —	
FM8 ・本文参照。	
PSP —	
PC88 ・TIME の代わりにTIME \$を用いる。	
N52 —	
PC6 —	
MLT —	
HC —	

タイマ割り込み先定義 ON TIME GOSUB

オン・タイム・ゴースブ

形式：ON TIME GOSUB 100

機能：タイマ割り込み時の処理ルーチン開始行を指定する。

- 特徴：
- 1 引数は行番号を示す。
 - 2 割り込み処理ルーチンの終わりには、RETURN 文をおかなければならない。
 - 3 インターバル・タイマ割り込みは ON INTERVAL 文で行う。

プログラム例：現在の時刻と終了時刻を入力し、時間を出力しながら終了時間に TIME OUT と出力してブザーを鳴らす。

```

10 CLS:ON TIME GOSUB 110
20 INPUT "PRESENT TIME";A$
30 INPUT "LAST TIME";B$
40 TIME$=A$
50 TIME B$
60 TIME ON
70 LOCATE 0,3:PRINT TIME$
80 GOTO 70
90 TIME OFF
100 END
110 LOCATE 0,3:PRINT TIME$
120 PRINT "TIME OUT"
130 BEEP1
140 FOR I=0 TO 200:NEXT
150 BEEP0
160 RETURN 90
    
```

実行例

```

PRESENT TIME? "15:01:00"
LAST TIME? "15:02:00"

15:02:00
TIME OUT

Ready
    
```

参照：TIME\$, TIME ON

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF₈₂ —

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 ・ 本文参照。

PSP —

PC88 ・ TIME の代わりに TIME \$ を用いる。

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ON TIME, 11.50 GOSUB 100
・ 最初の引数は時間を表す (11時50分)。指定した時間になると、指定した文番号 (100) のサブルーチンへとぶ。

X1Hu —

MB16 —

IBM55 —

PC100 ・ ON TIMER(n) GOSUB L
・ n は秒を表し 1 ~ 65535, L はラベルまたは行番号を示す。ほかは本文と同じ。

MSX ・ ON INTERVAL = M1 GOSUB M2
・ M1/60秒ごとにタイマ割り込みを行う。M2 は行番号である。

キーによる割り込み ON KEY GOSUB, OFF KEY

オン・キー・ゴー・サブ, オフ・キー

形式: ① ON KEY CHR\$&01, "Z" GOSUB

*省略・繰り返し可

キーの指定

1000, "E"

*省略・繰り返し可

飛び先ルーチンの指定

② OFF KEY

機能: ①一度この文が実行されると、以降において指定されたキーが押されたとき、指定したサブルーチンへ飛ぶ。

②その指定を解除する。

特徴: 1(a)この例では、アスキー・コードが01の (DEF) キーが押されると、サブルーチン1000へ飛ぶ。また、(Z) キーが押されるとライン・ラベル "E" からのサブルーチンへ飛ぶ。

(b)キーの指定の個数は、28個まで。

(c)キー指定で "ABC" のようにすると、最初の1文字 "A" が指定されたことになる。

(d)ON KEY で再度ほかの指定をすると、前の指定は全部無効になる。

(e)INPUT 文による待状態のときは、この割り込み機能は効かない。

(f)GOSUB の代わりに、GOTO を書くことも可能である。その場合は指定行へ単純にジャンプする。

(g)ON KEY を一度実行すると、次は無効になる。すなわち1回しかキーを押せない。ON KEY をその都度実行させる必要があるので注意。

(h)利用者のプログラムを中断させたいとき、(STOP) キーや、(BRAKE) キーを押すと、BASIC モードへもどってしまう。そうしたくないとき、たとえば、利用者のプログラムのある箇所へもどしたいような場合に便利である。

プログラム例: "DEF 1" キーを押すと 1 TRAPPED と印字し、実行を継続し、"Z" キーを押すと、Z TRAPPED と印字し停止する。

```

10 GO SUB "TRAP"
20 FOR I=1 TO 30
30 PRINT I
40 NEXT I
50 GO TO 20
60*"A":PRINT "1 TRAPPED"
70 GO SUB "TRAP"
80 RETURN
90*"B":PRINT "Z TRAPPED"
100 GO SUB "TRAP"
110 STOP
120 END
130*"TRAP":ON KEY CHR$ &01,"Z" GO SUB "A","B"
140 RETURN

```

実行例

```

1
2
3
4
5 1 TRAPPED
6
7
8 Z TRAPPED

```

参照: HELP ON, PEN ON, COM ON, STOP ON, TIME ON

キーによる割り込み ON KEY GOSUB, OFF KEY

オン・キー・ゴースブ, オフ・キー

APL —	が実行されると対応するルーチンへ飛ぶ。	MLT ・IF8 ₂ と同じ。	
TRS ₁ —	・KEY (n) STOP によって、あるキー n だけを上記の意味で無効にできる。	HC —	
CPM —		FP11 ・IF8 ₂ と同じ。ただし、ファンクション・キーは0～9。また、特定のキーに対し、有効にしたり無効にすることはできない。	
M223 —	レベル ₃ ① ON KEY (1) GOSUB 1000	SMC —	
MZK —	・機能はIF8 ₂ と同じであるが、一つの文で一つのキーとその飛び先を指定できるのみである。したがって複数行によって、複数のキーの飛び先の宣言をする。	MZ35 ・本文と同じ。	
CBM —	② KEY (1) OFF	X1Hu ・IF8 ₂ と同じ。	
PC8 —	・キー番号 (1) は省略できない。機能はIF8 ₂ と同じ。	MB16 ・レベル ₃ と同じ。ただし、キー番号は1～14。	
TRS ₁₁ —	③ KEY (1) ON	IBM55 ・MB16と同じ。	
PC3 ・本文参照。	④ KEY (1) STOP	PC100 ・レベル ₃ と同じ。なお、行番号にはラベル名も使用できる。	
IF8 ₂ ① ON KEY GOSUB 1000, 1100, 1200, 1300	・③④ともIF8 ₂ と同じ。キー番号 (1) は省略できない。	MSX ・IF8 ₂ と同じ。ただし、KEY (n) ON, KEY (n) OFF, KEY (n) STOP において (n) は省略できない。	
・機能は本文と同じであるが、キーはファンクション・キー1～10の場合に限定されている。飛び先の順番がファンクション・キーの番号順に一致する。GOTOは使えない。	C180 —	・このほか (STOP) キーを割り込みキーとして使うための ON STOP GOSUB, STOP がある。	
・この文は単なる飛び先の宣言であって③のKEY ON を実行してはじめて有効になる。	M243 —		
② KEY OFF	MZB —		
・機能は本文と同じであるが、KEY (n) OFF によってあるキー n だけを無効にすることもできる。	BUB ・IF8 ₂ と同じ。		
③ KEY ON	FM8 ・レベル ₃ と同じ。		
・機能は本文と異なり②で無効にしたキーは①によらず、この文で有効にする。また①だけでは有効にならず、この文を実行してはじめて有効になる。	PSP —		
・KEY (n) ON によってあるキー n だけを有効にできる。	PC88 ・IF8 ₂ と同じ。ただし、飛び先にラベルを使える。		
④ KEY STOP	・このほか (HELP) キー, (STOP) キーを割り込みキーとして使うための ON HELP GOSUB, HELP ON, ON STOP GOSUB, STOP ON などがある。		
・この文によってもキー割り込みを無効にできるが、無効の期間に押されたキーは記憶されていて、③	N52 —		
	PC6 —		

ライト・ペンによる割り込み許可, 禁止 PEN ON, PEN OFF, PEN STOP

ペン・オン, ペン・オフ, ペン・ストップ

- 形式: ① PEN ON
 ② PEN OFF
 ③ PEN STOP

機能: ライト・ペンが押された時に起こる, 割り込みを許可, 禁止する.

- 特徴: 1 PEN ON で割り込みを許可する. この文を実行した後ライト・ペンを押せば, ON PEN GOSUB で指定された処理ルーチンに飛ぶ.
 2 PEN OFF で, 割り込みを禁止することができる.
 3 PEN STOP で, 割り込みを停止することができる. この文を実行した後, ライト・ペンを押しても, 処理ルーチンへ飛ばないが, PEN ON とすれば, 前にライト・ペンを押していたことによって, 処理ルーチンに飛ぶ.
 4 プログラム終了時には, PEN OFF を実行する.

プログラム例: 主プログラムでは2から1000までの素数を計算し, プリンタに出力するプログラムである. このプログラムを実行中であっても, ライトペンで CRT 上に直線を描くことができる(実行例は ON PEN GOSUB を参照).

```

100 OPEN "0",#5,"LPT0:"
110 COLOR 7,1
120 ON PEN GOSUB 280
130 PEN ON
140 CLS
150 PRINT #5
160 N=2
170 PRINT #5,USING "####";2;3;
180 FOR I=5 TO 1000 STEP 2
190   FOR J=3 TO SQR(I) STEP 2
200     IF I MOD J = 0 THEN 230 ELSE NEXT
210   PRINT #5,USING "####";I;
220   N=N+1:IF N>9 THEN N=0:PRINT #5
230 NEXT I
240 PEN OFF
250 CLOSE
260 END
270 'LPEN ROUTINE
280 ST=ST+1
290 IF ST=1 THEN X=PEN(1):Y=PEN(2) ELSE ST=0
300 LINE (X,Y)-(PEN(1),PEN(2)), "*"
310 RETURN
  
```

参照: ON PEN GOSUB, PEN

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF₈₂ ・本文と同じ.

レベル₃ ・本文参照.

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 ・本文と同じ.

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu —

MB16 ・本文と同じ.

IBM55 —

PC100 —

MSX —

ライト・ペンによる割り込み ON PEN GOSUB

オン・ペン・ゴーサブ

形式: ON PEN GOSUB 70, 100

*省略・繰り返し可

機能: ライト・ペンが押され割り込みが起こった時、分岐する処理ルーチンの開始行を指定する。

- 特徴: 1 PEN ON のとき動作する。
 2 PEN 文で指定された画面上のエリア、(1~9) でライト・ペンが押された場合、そのエリアに対応した処理ルーチンに分岐する。最初の行番号は、エリア以外の入力の場合で、それに続く行番号はエリア 1, 2, ... の順に対応する。
 3 エリアは、1~9 まで定義できる。

プログラム例: 主プログラムでは 2 から 1000 までの素数を計算し、プリンタに出力するプログラムである。このプログラムを実行中であっても、ライト・ペンで CRT 上に直線を描くことができる (プログラム例は PEN ON を参照)。

実行例

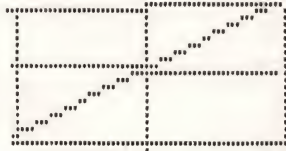
素数のプリンタ出力。

```

2 3 5 7 11 13 17 19 23 29
31 37 41 43 47 53 59 61 67 71
73 79 83 89 97 101 103 107 109 113
127 131 137 139 149 151 157 163 167 173
179 191 193 197 199 211 223 227 229
233 239 241 251 257 263 269 271 277 281
283 293 307 311 313 317 331 337 347 349
353 359 367 373 379 383 389 397 401 409
419 421 431 433 439 443 449 457 461 463
467 479 487 491 499 503 509 521 523 541
547 557 563 569 571 577 587 593 599 601
607 613 617 619 631 641 643 647 653 659
661 673 677 683 691 701 709 719 727 733
739 743 751 757 761 769 773 787 797 809
811 821 823 827 829 839 853 857 859 863
877 881 883 887 907 911 919 929 937 941
947 953 967 971 977 983 991 997

```

ライト・ペンによる CRT 出力。



参照: PEN ON, PEN

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF₈₂ ・ 本文と同じ。レベル₃ ・ 本文参照。

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 ・ 行番号にラベルも使える。

N52 —

PC6 —

MLT —

HC —

FPI1 —

SMC ・ 該当なし。割り込みによらない、ライト・ペン情報の読み取りは PEN READ 命令を用いる。

MZ35 —

X1Hu —

MB16 ・ 行番号は一つで分岐は起これない。
・ 特徴 2, 3 は使えない。

IBM55 —

PC100 —

MSX 該当なし。ただし、ジョイスティックのトリガ・ボタンにより割り込みを行う。ON STRIG GOSUB, ON STRIG ON/OFF /STOP がある。

ファンクション・キーの定義 KEY ファンクション

形式: KEY 2, "タテ"

機能: プログラマブル・ファンクション・キーの定義。

- 特徴:
- 1 KEYの指定は、キー番号、(,), 文字列の順である。
 - 2 シフト・モードを含めて、10のプログラマブル・コードを記憶できる。キー番号は、1から10まで。
 - 3 最大15文字までの文字列、およびコントロール文字を各ファンクション・キーにプログラムできる。キーボードから入力できない文字は、CHR\$関数を(+)でつないで用いる。
 - 4 KEY LIST 文によって、ファンクション・キーの内容を画面に表示できる。

実行例: ファンクション・キーを定義する。

```

1 1,"SORT"
2 2,"FUNC"
3 3,"PLAY"
4 4,"EDIT"
5,"LIST"

```

```

SORT  FUNC  PLAY  EDIT  LIST

```

参照: KEY LIST

APL —

TRS₁ —

CPM —

M223 ・OPKEYを用いる。キーの数は8個、31文字まで入力できる。

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 DFK 2, "タテ"
・最大12文字で、20定義できる。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・SFUNCを用いる。

M243 ・FUNCを用いる。

MZB DEF KEY (1)=LIST ①
・デファイナブル・ファンクション・キー1に(LIST)-(CR)キーの機能を定義。
・すべてのキーの文字数の和は256文字を越えられない。
・ユリティリティ・プログラムを定義後走らせないと定義されない。

BUB ・ファンクション・キーの数は8個。
・(LABEL) キーによってもキーの内容が見られる。

FM8 ・本文と同じ。

PSP —

PC88 ・本文と同じ。

N52 ・特徴1, 3は同じ。
・キー番号はキーボード上のPFキーの番号に対応し、1~10の値でなければならない。

PC6 ・8文字まで入力できる。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・キー番号は0~9。

SMC DEF KEY 2, "タテ"
・最大16文字まで定義できる。

MZ35 ・PC3と同じ。

X1Hu ・本文と同じ。
・DEF KEY も同じ使い方。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。なお、ファンクション・キーのほかにもユーザが定義できるキーがある。

MSX ・本文と同じ。

ファンクション・キーの内容表示 KEY LIST

キー・リスト

形式：KEY LIST

機能：ファンクション・キーの内容を画面に表示する。

特徴：1 画面の最終行には、ファンクション・キーの内容の一部（6文字）しか表示できないが、このコマンドによりそのすべてが表示できる。

実行例：キー・リストを出力してみる。

KEY LIST

```
load      save
auto      key
go to     print
list      edit
run       cont
Ok
```

参照：KEY ファンクション

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP` —

PC88 ・本文と同じ。

N52 —

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。
・KEYLLIST で内蔵マイクロ・プリンタへ出力される。

FP11 ・本文と同じ。

SMC ・本文と同じ。

MZ35 LIST DFK

X1Hu ・本文と同じ。
・KLIST も同じ使い方。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

ファンクション・キーの内容のセーブとロード KSAVE, KLOAD

キー・セーブ, キー・ロード

形式: ① KSAVE "AFILE"

② KLOAD "BFILE"

ファイル名

機能: ①ファンクション・キーの定義内容を, ファイル名を指定して, ディスクに書き込む.

②ディスクにある指定されたファイルを読み込み, その内容により, ファンクション・キーを再定義する.

特徴: 1 KLOAD で指定するファイルは, 先に KSAVE を用いてセーブされたもの.

2 ファンクション・キーを幾通りにも定義する場合, この命令符を用いると便利である.

実行例: KEY1へ"INPUT", KEY2へ"LOAD"を定義して,
ファイル名"AFILE"というファイルへ格納する.

```
>LIST
  10 DFK 1,"INPUT"
  20 DFK 2,"LOAD"
  30 KSAVE "AFILE:A1"
  40 END
```

```
>RUN
READY
>
```

参照: KEY ファンクション

APL —

TRS_I —

CPM —

M223 ・ CLIで実行できる.
OPKEYを実行する. 表示はL, 終了はE, 1~8の数字を入力すると設定を変更できる.

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 ・ 本文参照.

IF8₂ —レベル₃ —C180 ・ ディスクへの書き込みは,
ユーティリティ FUNCTによって
行われ, BASIC では作成できない.
・ ディスク上のファイルは,
CFUNC AFILE, FK/3で参照
できる.

M243 ・ M223 と同じ.

MZB DEF KEY(n)=LIST↵
・ 表示は, KLIST で行える.

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 ・ PC3 と同じ.

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

時間監視 SLEEP

スリープ

形式：SLEEP 100

機能：指定時間の間、プログラムの実行を停止する。

- 特徴：1 時間の指定は単精度の算術式で行い、算術式の整数部のみが有効。
 2 指定時間の単位は秒。
 3 指定時間は0～16383秒の範囲。
 4 正確に、かつ簡単に待ち時間を設定できるので便利である。

プログラム例：10秒ごとに“BASIC”と出力させる。

```
10 OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20 FOR I = 1 TO 3
30 PRINT #1, "BASIC"
40 SLEEP 10
50 NEXT I
60 CLOSE 1
70 END
```

実行例

```
BASIC
BASIC
BASIC
```

参照：TIME\$

APL —

TRS_I —

CPM —

M223 ・本文参照。

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ PAUSE 100
 ・機能は本文と同じ。ただし、
 PAUSE 文の実行中でも (CAN)
 キーは有効。

レベル₃ —

C180 WAIT DELAY 100
 ・機能は本文と同じ。
 WAIT TIME "11:22:33"
 ・指定した時刻まで、プログラムの
 実行を停止する。

M243 ・本文と同じ。

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC PAUSE 100
 ・指定数値は0以上255以下で、指
 定時間は、その10分の1である。

MZ35 WAIT X
 ・Xは待機時間（単位0.1秒）。

X1H_u PAUSE 100
 ・引数は0～255（1単位約0.1秒）。

MB16 —

IBM55 —

PC100 —

MSX —

時間監視 WAIT 時間

ウェイト

形式: WAIT 100

機能: 時間制限付きのキーボードからの入力。

- 特徴:
- 1 時間の指定は単精度の算術式で行い、算術式の整数部のみが有効。
 - 2 指定時間以内に入力がないときは、内部処理可能なエラー(ERR 71)が発生する。
 - 3 制限時間の単位は秒。
 - 4 制限時間は 0 ~ 16383 秒の範囲。
 - 5 INPUT 文と対にして用いる。
 - 6 WAIT, INPUT 文が実行された後に、WAIT 0 で時間のリセットが必要。

プログラム例: メッセージに答えるまで、10秒ごとにメッセージを出
力する。

```

10      OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20      ON ERROR GOTO 100
30      WAIT 10
40      PRINT #1, "STOP PROGRAM ? YES(Y) or NO(N) "
50      INPUT AS$
60      WAIT 0
70      IF AS$ = "Y" THEN
:         PRINT #1, " *** PROGRAM STOP *** "
:      ELSE
:         GOTO 30
80      CLOSE 1
90      END
100     IF ERR = 71 THEN
:         PRINT #1,
:         RESUME 30
NO END MARK

```

実行例

```

STOP PROGRAM ? YES(Y) or NO(N)
STOP PROGRAM ? YES(Y) or NO(N)
STOP PROGRAM ? YES(Y) or NO(N)
*** PROGRAM STOP ***

```

参照: TIME\$, SLEEP

APL —

TRS_I —

CPM —

M223 ・ 本文参照。

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —レベル₃ INPUT WAIT 70 ; 100

"INPUT A" ; A1, A2

- ・ 指定した制限時間以内に入力がない
ければ、指定した行番号の行に制
御が移る。
- ・ 制限時間は1~255秒の範囲。

C180 WAIT EVENT FUNKEY
TIMEOUT 100

- ・ 指定した制限時間以内にファンク
ション・キーの入力がなければ、
エラーとなる。
- ・ 制限時間は0~86400秒の範囲。
- ・ 制限時間を"HH:MM:SS"の型
式の時刻でも指定することができ
る。

M243 ・ 本文と同じ。

MZB —

BUB —

FM8 —

PSP —

PC88 INPUT WAIT 100,

"INPUT A" ; A1, A2

- ・ この文の後にマルチ・ステートメ
ントとして文をつけたとき、制限
時間以内に入力が行われたときの
み後の文を実行し、入力がなかつ
たときは次の行へ実行が移る。
- ・ 制限時間の単位は 0.1 秒。

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

入力データの監視 WAIT データ

ウェイト

形式: WAIT 1, 2, 3

省略可

機能: 指定したポートからの入力を監視し、入力データの条件が満足されるまで実行を停止する。

- 特徴: 1 指定した入力ポートのビット・パターンが指定した状態になるまでプログラムの実行が中断される。
 2 1 番目の引数でポート番号を指定する。
 3 指定したポートから読みこんだデータと (3 番目の引数) との XOR の結果と (2 番目の引数) の AND がとられ、その結果が 0 ならばもう一度ポートの状態を読み込み同じ操作を繰り返す。結果が 0 でなければ次の文へ実行が移る。
 4 3 番目の引数が省略されたときは 0 とみなされる。
 5 WAIT 文の実行により無限ループになったときは、リセットしなければならない。

プログラム例: テンキーの 1 から 6 を入力し、入力と発生させた乱数とが等しいかを調べる。

```
10 PRINT "PLEASE PUSH TEN-KEY (1-6)"
20 WAIT 0,126,255
30 INPUT X%
40 Y%=RND(1)*6+1
50 IF X%=Y% THEN PRINT "ATARI":END
   ELSE PRINT "HAZURE"
60 GOTO10
```

実行例

```
PLEASE PUSH TEN-KEY (1-6)
? 5
HAZURE
PLEASE PUSH TEN-KEY (1-6)
? 2
HAZURE
PLEASE PUSH TEN-KEY (1-6)
? 3
ATARI
```

参照: IN, OUT, SLEEP

APL ・本文と同じ。

TRS_I —

CPM ・本文と同じ。

M223 —

MZK —

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ ・引数に小数が与えられたときは、小数部は四捨五入される。小数部は四捨五入される。
 ・WAIT 文の実行中でも (CAN) キーは有効。

レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP WAIT #1, &H02, &H03
 ・機能は本文と同じ。

PC88 ・本文と同じ。

N52 —

PC6 —

MLT ・本文と同じ。

HC —

FP11 —

SMC —

MZ35 —

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。ただし、特徴5 は BRAKE でもよい。

PC100 ・本文と同じ。

MSX ・本文と同じ。

プリンタ・ヘッドの位置表示 LPOS

エル・ポジション

形式：LPOS (A)

機能：プリンタ・バッファの現在のヘッドの位置を与える関数。

特徴：1 引数はダミーである。

プログラム例：疑問符(?)の並びの後尾位置を与える。

```

10 FOR I=0 TO 30 STEP 5
20 LPRINT STRING$(I,"?");
30 L1=LPOS(34)
40 LPRINT " LPOS=";L1
50 NEXT I
60 END

```

実行例

```

LPOS= 0
????? LPOS= 5
??????????? LPOS= 10
??????????????? LPOS= 15
??????????????????? LPOS= 20
??????????????????????? LPOS= 25
??????????????????????????? LPOS= 30

```

参照：POS

APL —

TRS_I —

CPM ・本文と同じ。

M223 POS (M)
 ・Mは、オープンされた、プリンタ
 装置番号。

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} —

PC3 —

IF8₂ ・本文と同じ。

レベル₃ POS (M)
 ・1 ≤ M ≤ 16で、ファイル番号Mで
 OPENされたプリンタのヘッドの
 水平位置を与える。

C180 ・該当なし。LPOSは機能
 が異なる。

M243 ・M223と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・レベル₃と同じ。

PSP —

PC88 ・本文と同じ。

N52 —

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・レベル₃と同じ。

FP11 —

SMC —

MZ35 —

X1Hu ・本文と同じ。

MB16 ・引数はプリンタ番号で、
 1から4までの整数。

IBM55 ・引数はプリンタ番号で、
 1から3までの整数。

PC100 ・本文と同じ。

MSX ・本文と同じ。

紙送り PAGE

ページ

形式：PAGE

機能：プログラム・リストをプリンタへ出力するときに、プリンタの次ページの先頭まで紙送りをする。

- 特徴：1 プリンタによるプログラム・リストを、見やすくするときに使うと便利である。
2 特に命令がない場合は、改頁のコード(&H0C)を出力すればよい。
LPRINT CHR\$(&H0C);

実行例：PAGE を用いてプリンタに出力してみる。

1 ページ目

```
10      FOR I = 1 TO 100
20          PRINT I , I*I , SQR(I)
30      PAGE
```

2 ページ目

```
40      NEXT I
50      PAGE
```

3 ページ目

```
60      END
```

参照：

APL —

TRS_I —

CPM —

M223 ・本文参照。

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 —

M243 ・本文と同じ。

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 LIST PRINT "E1"
・E1はタイトル。

X1Hu —

MB16 —

IBM55 —

PC100 ・キャラクタ・デバイス(プリンタなど)にコントロール文字列を送る命令にIOCTL, その確認の関数としてIOCTL\$がある。

MSX —

バブル・メモリのアクセス BUBR, BUBW

バブル・リード, バブル・ライト

形式: ① BUBR M, N, E\$

② BUBW M, N, E\$

機能: バブル・メモリのリード, ライトを行う。

- 特徴: 1 Mはユニット¹⁾番号で, 0 または 1 を指定する。
 2 Nはページ²⁾番号で, 1 から 1024 の間で指定する。
 3 E\$ は入出力バッファであり, 文字変数名あるいは配列名で指定する。
 4 アクセス³⁾の高速性が望まれ, メイン・メモリでは足りない中容量ファイルに適す。

用語: ¹⁾ユニット 通常, 物理的に識別される装置のこと。
²⁾ページ メモリを割り当てたり, プログラムを制御セクションに区分するために使用される主記憶容量の標準量, この場合は一回にアクセスできる量を表す。
³⁾アクセス 記憶装置からデータを読み取ったり, そこにデータを書き込んだりすること。

プログラム例: 0 番ユニット, 第 1 ページにキーボードより入力した文字列を書き込み, 読み出して確認する。

```
10 INPUT E1$
20 BUBW 0,1,E1$
30 BUBR 0,1,E2$
40 PRINT E2$
50 END
```

実行例

? ABCDEFG

ABCDEFGF

Ready

参照:

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 ・ 本文参照。

PSP —

PC88 —

N52 ・ 該当なし。ただし, 同種の機能として, バックアップ・メモリがある。
 ・ BUMI\$ および BUMO\$ によりディスク上のユーザ領域を仮想メモリとして使用できる。

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

平方根 SQR

スクエア・ルート

形式：SQR (A)

機能：A の平方根を求める関数。

特徴：1 A は負の数であってはならない。
2 演算は単精度で行われる。

プログラム例：SQR (X) を、X=1 から X=10 まで求める。

```
10 FOR I=1 TO 10
20 LPRINT "SQR(";I;")=",SQR(I)
30 NEXT I
40 END
```

実行例

```
SQR ( 1 ) =      1
SQR ( 2 ) =    1.41421
SQR ( 3 ) =    1.73205
SQR ( 4 ) =      2
SQR ( 5 ) =    2.23607
SQR ( 6 ) =    2.44949
SQR ( 7 ) =    2.64575
SQR ( 8 ) =    2.82843
SQR ( 9 ) =      3
SQR (10 ) =    3.16228
```

参照：

APL ・本文と同じ。(参考：演算は、 $A^{0.5}$ よりも速い)

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。ただし、A が符号を伴わない数値定数、数値変数の場合や関数を伴う場合、カッコで囲む必要はない。

IF8₂ ・本文と同じ。ただし、A が倍精度数なら倍精度として扱われる。

レベル₃ ・本文と同じ。

C180 ・本文と同じ。

M243 ・本文と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・引数の型により倍々精度まで求められる。

SMC ・本文と同じ。

MZ35 ・PC3 と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。
・演算は倍精度で行われる。

MSX ・PC100 と同じ。

三角関数 SIN, COS, TAN

サイン, コサイン, タンジェント

形式: SIN (A), COS (A), TAN (A)

機能: A の単位をラジアンとした場合, 三角関数の値を求める関数.

- 特徴: 1 演算は単精度で行われる.
 2 引数は「度」でなく「ラジアン」で与えなくてはいけないので
 注意が必要(度からラジアンへの変換は, $\frac{\pi}{180}$ を乗ればよい).
 3 TAN (A) でオーバーフロー¹⁾した場合はエラーが起き, 実行は停止する.

用語: ¹⁾オーバーフロー 計算結果が, 計算機で扱える数値の最大値を超えることをいう.プログラム例: SIN (0), SIN ($\frac{\pi}{2}$), SIN ($\frac{\pi}{3}$), SIN ($\frac{3}{4}\pi$) を求める.

```
10 PAI=3.14159
20 FOR I=0 TO 3
30 A1=PAI/4*I
40 A2=SIN(A1)
50 LPRINT "SIN(";A1;")=";A2
60 NEXT I
70 END
```

実行例

```
SIN( 0 )= 0
SIN( .785398 )= .707106
SIN( 1.5708 )= 1
SIN( 2.35619 )= .707108
```

参照: ATN

APL ・ 本文と同じ.

TRS₁ ・ 本文と同じ.CPM ・ オーバーフローが生じて
もプログラムの実行は続く.

M223 ・ 本文と同じ.

MZK ・ 本文と同じ.

CBM ・ 本文と同じ.

PC8 ・ 本文参照.

TRS₁₁ ・ 本文と同じ.

PC3 ・ 本文と同じ. ただし, 引
数 A の単位は, 度, ラジアンのど
ちらでも使える (プログラムの最
初に DEG または RAD 文で指定
しておく). A が符号を伴わない
数値定数, 数値変数の場合や関数
を伴う場合, カッコで囲む必要は
ない.

IF8₂ ・ A が倍精度数なら倍精度
として扱われる.
・ オーバーフローが生じててもメッセ
ージを出力するだけで, プログラム
の実行は続く.

レベル₃ ・ 本文と同じ.

C180 ・ 本文と同じ.

M243 ・ 本文と同じ.

MZB ・ 本文と同じ.

BUB ・ 本文と同じ.

FM8 ・ 本文と同じ.

PSP ・ 本文と同じ.

PC88 ・ 本文と同じ.

N52 ・ 本文と同じ.

PC6 ・ 本文と同じ.

MLT ・ オーバーフローが生じた場
合, 表現可能な最も大きい符号付
の数値を返してプログラムを継続
する. メッセージも出力される.

HC ・ 本文と同じ.

FP11 ・ 特徴は ATN の項を参照.
・ 指定できる角度の大きさに限度が
あるので注意が必要.
・ 関連して, 双曲線関数 HSN,
HCS, HTN もある.

SMC ・ 本文と同じ.

MZ35 ・ PC3 と同じ.

X1Hu ・ 本文と同じ.
RAD (10)
・ 度をラジアンに変換する.

MB16 ・ 本文と同じ.

IBM55 ・ 本文と同じ. ただし, 特
徴 3 はない.

PC100 ・ 本文と同じ.
・ 演算は倍精度で行われる.

MSX ・ PC100 と同じ.

三角関数逆正接 ATN

アーク・タンジェント

形式：ATN (A)

機能：A の単位をラジアンとした場合、三角関数逆正接の値を求める関数。

特徴：1 演算は単精度で行われる。

2 求められる値は、 $-\frac{\pi}{2}$ から $\frac{\pi}{2}$ の間の値である。

プログラム例： $\text{TAN}(\theta) = -0.25$ となるような θ を $-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$ の範囲で求める。

```
10 A1=-.25
20 A2=ATN(A1)
30 LPRINT "ATN(" ; A1 ; ")=" ; A2
40 END
```

実行例

ATN(-.25)=-.244979

参照：TAN, SIN, COS

APL ・ 本文と同じ。

TRS_I ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ 本文と同じ。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ DEG または RAD 文で度またはラジアンを指定できる。
・ A が符号を伴わない数値変数、数値定数の場合や、関数を伴う場合、カッコで囲む必要はない。
・ 逆正弦 ASN および逆余弦 ACS もある。

IF8₂ ・ A が倍精度数なら倍精度として扱われる。

レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。

M243 ・ 本文と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 —

MLT ・ 本文と同じ。

HC ・ 本文と同じ

FP11 ・ 引数の型により倍精度まで求められる。
・ カッコは省略できない。
・ 単位は ANGLE 文により変更できる（電源投入時は度に変更されている）。DEG（度）、RAD（ラジアン）、GRAD（グラディアン）がある。
・ 逆正弦 ASN および逆余弦 ACS もある。
・ さらに、逆双曲線関数 AHS、AHC、AHT もある。

SMC ・ 本文と同じ。

MZ35 ・ PC3 と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。
・ 演算は倍精度で行われる。

MSX ・ PC100 と同じ。

アングル

角度モードの指定 ANGLE

形式: ANGLE 1

機能: 三角関数の角度モードを指定する。

特徴: 1 角度指定は 0 から 2 で, 0 は DEG (度), 1 は RAD (ラディアン), 2 は GRAD (グラディアン) である。数式が使える, 小数部は切り捨てられる。
 2 電源 ON 時は ANGLE0 に設定される。

プログラム例: SIN (30°) を各種の単位指定で求める。

```
10 ANGLE 0
20 PRINT SIN(30);
30 ANGLE 1
40 PRINT SIN(PI/6);
50 ANGLE 2
60 PRINT SIN(100/3)
70 END
```

実行例

0.5 0.5 0.5

参照:

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 ・ MZ35 と同じ。

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 ・ 本文参照。

SMC —

MZ35 ・ DEG で度, RAD でラディアン
 の指定が可能である。
 ・ 電源 ON 時は DEG に設定される。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

指数関数 EXP

イクスポネント

形式：EXP (A)

機能：自然対数の底 e を A 乗した値を求める関数。

特徴：1 A には倍精度の値も許されるが、演算は単精度で行われる。

プログラム例：e^x を x=1, 2, 3, 4, 5 のそれぞれについて求める。

```
10 FOR I= 1 TO 5
20 A=EXP(I)
30 LPRINT "EXP(" ; I ; ")=" ; A
40 NEXT I
50 END
```

実行例

```
EXP( 1 )= 2.71828
EXP( 2 )= 7.38906
EXP( 3 )= 20.0855
EXP( 4 )= 54.5982
EXP( 5 )= 148.413
```

参照：LOG

APL ・本文と同じ。

TRS_I ・本文と同じ。

CPM ・本文と同じ。ただし、
A ≤ 87.3365 を満たさなければならない。

M223 ・本文と同じ。

MZK ・本文と同じ。

CBM ・本文と同じ。ただし、
A ≤ 88.02969191 を満たさなければならない。

PC8 ・本文参照。ただし、
A < 87.3366 を満たさなければならない。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。ただし、A
が符号を伴わない数値定数、数値
変数の場合や関数を伴う場合、カ
ッコで囲む必要はない。

IF8₂ ・本文と同じ。ただし、
A ≤ 88.0297 を満たさなければならない。

レベル₃ ・本文と同じ。

C180 ・本文と同じ。

M243 ・本文と同じ。

MZB ・本文と同じ。

BUB ・A ≤ 88.0297。

FM8 ・A < 88.0297。

PSP ・本文と同じ。

PC88 ・A ≤ 87.33655。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・IF8₂ と同じ。ただし、
over flow したときは最大値が入
る。

HC ・A ≤ 88.02969 を満たさ
なければならない。

FP11 ・A < 230。また、A が倍
精度の時、結果は倍精度で得られ
る。

SMC ・A ≤ 147.36544595161 を
満たしていなければならない。

MZ35 ・PC3 と同じ。

X1Hu ・HC と同じ。

MB16 ・A < 88.02968。

IBM55 ・HC と同じ。

PC100 ・本文と同じ。
・演算は倍精度で行われる。

MSX ・PC100 と同じ。

常用対数 LOG₁₀

形式：LOG (A)

機能：A の常用対数を求める関数。

特徴：1 A の値は正でなくてはならない。
 2 LOG (A) / LOG (10) で常用対数のない機種でも求められる。

プログラム例：10 から 100 までの常用対数の値を求める。

```

10 OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20 FOR I = 10 TO 100 STEP 10
30 LET A = LOG(I)
40 PRINT #1, "LOG(" ; I ; ")=" ; A
50 NEXT I
60 CLOSE 1
70 END

```

実行例
 実行例
 実行例

LOG(10) = 1	LOG(60) = 1.77815
LOG(20) = 1.30103	LOG(70) = 1.8451
LOG(30) = 1.47712	LOG(80) = 1.90309
LOG(40) = 1.60206	LOG(90) = 1.95424
LOG(50) = 1.69897	LOG(100) = 2

参照：LOG。

APL —

TRS_I —

CPM —

M223 ・ 本文参照。

MZK ・ 本文と同じ。

CBM —

PC8 —

TRS_{II} —

PC3 ・ 本文と同じ。ただし、A
 が符号を伴わない数値定数、数値
 変数の場合や関数を伴う場合は、
 カッコで囲む必要はない。

IF8₂ —レベル₃ —

C180 —

M243 ・ 本文と同じ。

MZB ・ 本文と同じ。

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 LGT (A)
 ・ 引数の型により倍精度まで求めら
 れる。

SMC ・ 本文と同じ。

MZ35 ・ PC3 と同じ。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

自然対数 LOGe

ロガリズム

形式: LOG (A)

機能: A の自然対数を求める関数。

特徴: 1 A の値は正でなくてはならない。

プログラム例: $\log_e x$ を $x=1, 2, 3, 4, 5$ のそれぞれについて求める。

```
10 FOR I=1 TO 5
20 A=LOG(I)
30 LPRINT "LOG(";I;")=";A
40 NEXT I
50 END
```

実行例

```
LOG( 1 ) = 0
LOG( 2 ) = .693147
LOG( 3 ) = 1.09861
LOG( 4 ) = 1.38629
LOG( 5 ) = 1.60944
```

参照: LOG₁₀

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 LN (A)
・ 機能は本文と同じ。

MZK ・ M223 と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS₁₁ ・ 本文と同じ。PC3 ・ M223 と同じ。ただし、A
が符号を伴わない数値定数、数値
変数の場合や関数を伴う場合、カ
ッコで囲む必要はない。IF8₂ ・ 本文と同じ。ただし、A
に倍精度数も使える。レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。

M243 ・ M223 と同じ。

MZB ・ M223 と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ 引数の型により倍精度ま
で求められる。

SMC ・ M223 と同じ。

MZ35 ・ PC3 と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。
・ 演算は倍精度で行われる。

MSX ・ PC100 と同じ。

絶対値 ABS

アブソリュート

形式: ABS (A)

機能: 引数 A の絶対値を与える関数.

特徴: 1 演算は単精度で行われる.

プログラム例: A1 の絶対値を A2 へ代入する.

```

10 A1=-1.234
20 A2=ABS(A1)
30 LPRINT "A1=";A1,"A2=";A2
40 END

```

実行例

```

A1=-1.234      A2= 1.234

```

参照:

APL ・ 本文と同じ.

TRS₁ ・ $-1.7 \times 10^{38} \leq A \leq 1.7 \times 10^{38}$ を満たさねばならない.

CPM ・ 本文と同じ.

M223 ・ ABS#(C#) で高精度演算もおこなえる.

MZK ・ 本文と同じ.

CBM ・ 本文と同じ.

PC8 ・ 本文参照.

TRS₁₁ ・ 本文と同じ.

PC3 ・ A が負符号を伴わない数値定数、数値変数の場合や関数を伴う場合、カッコで囲む必要はない.

IF8₂ ・ 本文と同じ.レベル₃ ・ 本文と同じ.

C180 ・ 本文と同じ.

M243 ・ M223 と同じ.

MZB ・ 本文と同じ.

BUB ・ 本文と同じ.

FM8 ・ 倍精度も可.

PSP ・ 本文と同じ.

PC88 ・ 本文と同じ.

N52 ・ 本文と同じ.

PC6 ・ 本文と同じ.

MLT ・ 本文と同じ.

HC ・ 本文と同じ.

FP11 ・ 引数の型により倍々精度まで求められる.

SMC ・ 本文と同じ.

MZ35 ・ PC3 と同じ.

X1Hu ・ 本文と同じ.

MB16 ・ 本文と同じ、ただし、演算の引数の型により倍精度まで行える.

IBM55 ・ 本文と同じ.

PC100 ・ 本文と同じ.
・ 演算は倍精度で行われる.

MSX ・ PC100 と同じ.

剰余 MOD

モジュール

形式：10 MOD 3

機能：剰余（割り算の余り）の計算をする。上記の例では1となる（ $10 \div 3 = 3$ 余り 1）。

特徴：1 剰余演算の演算順位は整数除算の次である。
 2 実数はすべて整数に切り捨てられる。
 3 $X \text{ MOD } Y = X - Y * (X / Y)$ で代用できる。

プログラム例：整数除算 $10 \div 4 = 2$ 余り 2 を計算し印字する。

```
10 M1=10
20 M2=4
30 N1=M1 / M2
40 N2=M1 MOD M2
50 LPRINT"SYO=";N1
60 LPRINT"AMARI=";N2
70 END
```

実行例

```
SYO= 2
AMARI= 2
```

参照：算術演算子（その1）

APL —

TRS_I —

CPM ・実数は四捨五入される。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・CPMと同じ。

PC3 ・ $A = \text{MOD}(10, 3)$ のように数値関数によって行う。
 ・上記の例ではAに1が代入される。

IF8₂ ・本文と同じ。ただし、扱われる数値は四捨五入される。

レベル₃ ・本文と同じ。

C180 ・PC3と同様に数値関数によって行う。ただし、REM関数も同様の働きをする。
 ・関数の意味は次のとおり。
 $\text{MOD}(3.2, 2) = 1.2$
 $\text{MOD}(X, Y) = X - Y * \text{INT}(X/Y)$
 $\text{REM}(X, Y) = X - Y * \text{IP}(X/Y)$

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・IF8₂と同じ。

PSP —

PC88 ・IF8₂と同じ。

N52 ・CPMと同じ。

PC6 —

MLT ・ $A \text{ MOD } B$ で、Aは整数値でなくてはいけない。Bは実数でよいが、小数点以下を四捨五入される。

HC —

FP11 ・演算順位は*、/、%と同等。

SMC ・IF8₂と同じ。

MZ35 ・PC3と同じ。

XIHu ・本文と同じ。

MB16 ・MODの前後には空白をあける。
 ・実数は四捨五入される。

IBM55 ・CPMと同じ。

PC100 ・CPMと同じ。

MSX ・本文と同じ。

符号関数 SGN

形式：SGN (A)

機能：Aの符号を求める関数。

特徴：1 Aが負のとき-1, 0のとき0, 正のとき1になる。
 2 ON~GOTO~文と併用すると便利である。

プログラム例：Iの値によって、飛び先を振り分ける。

```

10 FOR I=-3 TO 3
20 ON SGN(I)+2 GOTO 30,50,70
30 LPRINT "I=";I,"-1"
40 GOTO 90
50 LPRINT "I=";I,"0"
60 GOTO 90
70 LPRINT "I=";I,"1"
80 GOTO 90
90 NEXT I
100 END

```

実行例

I=-3	-1
I=-2	-1
I=-1	-1
I= 0	0
I= 1	1
I= 2	1
I= 3	1

参照：

APL ・本文と同じ。

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。ただし、
 SGN# (C#) で高精度演算もできる。

MZK ・本文と同じ。

CBM ・本文と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・本文と同じ。ただし、A
 が符号を伴わない数値定数、数値
 変数の場合や関数を伴う場合、カ
 ッコで囲む必要はない。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 ・本文と同じ。

M243 ・M223 と同じ。

MZB ・本文と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・引数の型により倍々精度
 まで求められる。

SMC ・本文と同じ。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

丸め ROUND

ラウンド

形式：ROUND (A1, A2)

省略可

機能：小数 A1 から A2 で指定された有効けた数をとる。四捨五入は A2 + 1 けた目を行う。

- 特徴：
- 1 A2 は省略できる。
 - 2 A2 を省略すると、0 とみなすので小数第 1 位を四捨五入し、整数になる。
 - 3 有効けた数を指定するときに使う。
 - 4 A2 = 0 のとき、INT (A1 + 0.5) で代用できる。
A2 = 1 のときは、INT (A1 * 10 + 0.5) / 10。
A2 > 1 のときも同様。

プログラム例：1 円周率を小数第四位を四捨五入して、小数第三位まで求める。
2 小数第一位を四捨五入して整数にする。

```
10 A=3.141592
20 A1=ROUND(A,3)
30 A2=ROUND(A)
40 PRINT "ROUND(3.141592,3)=";A1
50 PRINT "ROUND(3.141592)=";A2
60 END
```

実行例

```
ROUND(3.141592,3)= 3.142
ROUND(3.141592)= 3
```

参照：NORMAL, FIXED, FLOAT

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 ・表示のとき、FIXED m, n, FLOAT m, n のようにして指定できる、その項参照。

IF8₂ —

レベル₃ —

C180 ・本文参照。

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 ROUND (A1, A2)

- ・指定は引数、桁位置の順。
- ・10 の A2 乗の位置の数字を四捨五入。
- ・引数の型により倍々精度まで求められる。

SMC —

MZ35 ・PC3 と同じ。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

小数部の取り出し FRAC

フラクション

形式: FRAC (A)

機能: 数字の小数部を取り出す。

- 特徴: 1 整数部の桁数が大きい数は、小数部を取り出すと誤差が大きくなる。
 2 $FRAC(A) = A - FIX(A)$ で代用できる。

実行例: 与えられた数 1.234 の小数部 0.234 を取り出す。

```
PRINT FRAC(1.234)
.234
OK
```

参照:

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 —

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 ・ 本文と同じ。

SMC —

MZ35 —

X1Hu ・ 本文参照。

MB16 —

IBM55 —

PC100 —

MSX —

最大値, 最小値 MAX, MIN

マキシマム, ミニマム

形式: ① MAX (A1, A2) ② MIN (A1, A2)

機能: A1 と A2 の二つの値のうち大きい方または小さい方の値を得る。

特徴: 1 IF 文を使わなくても大きい値または小さい方がわかるので便利である。

2 $MAX(A1, A2) = (A1 + A2 + ABS(A1 - A2)) / 2$

$MIN(A1, A2) = (A1 + A2 - ABS(A1 - A2)) / 2$

で代用できる。または、

$MAX(A1, A2) = -A1 * (A1 >= A2)$

$-A2 * (A1 < A2)$

$MIN(A1, A2) = -A1 * (A1 <= A2)$

$-A2 * (A1 > A2)$

プログラム例: 13.4 と 14.5 を比較して大きい方を判定する。

```
10 A=MAX(13.4,14.5)
```

```
20 PRINT "MAX=";A
```

```
30 END
```

実行例

MAX= 14.5

参照:

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 ・本文参照。

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 ・該当なし。関連して、基

本統計量を算出する17種の統計関数をもつ。

SMC —

MZ35 ・該当なし。関連して、数値配列中で最大(小)値が入っている要素の位置を求める、MAXS (MINS) 命令がある。

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

乱数の発生 RND

ランダム

<p>形式：RND (A)</p> <p>機能：0 から 1 までの単精度の乱数を求める関数。</p> <p>特徴：</p> <ol style="list-style-type: none"> 1 A が負の場合は A の値に従属した新系列の乱数を先頭より発生する。 2 A が 0 の場合は、一つ前に発生した乱数を発生する。 3 通常は正の数を引数として使い、同系列の乱数を順次発生させる。 4 ゲームやシミュレーションのとき必要である。 	<p>APL ・本文と同じ。ただし、A が正のとき新系列の乱数を、0 のとき一つ前に発生した乱数を、負のとき同系列の乱数を発生する。</p> <p>TRS_I ・RND (0) は 0 から 1 までの乱数を、RND (M) は 0 から M までの乱数を発生する。</p> <p>CPM ・本文と同じ。ただし、RND だけでも可。このとき 3 と同じになる。</p> <p>M223 RND ・機能は本文と同じ。</p> <p>MZK ・引数に 0 または負の整数を与えると乱数発生 of イニシャライズが行われ、正の整数を与えると同系列の乱数を順次発生する。</p> <p>CBM ・本文と同じ。</p>	<p>FM8 ・CPM と同じ。</p> <p>PSP ・M223 と同じ。ただし、IRND (N) は $1 \leq N \leq 9999$ であり、1 から N までの整数の乱数を発生する。</p> <p>PC88 ・CPM と同じ。</p> <p>N52 ・本文と同じ。</p> <p>PC6 ・本文と同じ。</p> <p>MLT ・本文と同じ。</p> <p>HC ・本文と同じ。</p> <p>FP11 ・本文と同じ。</p> <p>SMC ・(A) を省略できる。 ・IRND (N) で 0 以上 N 以下の乱数を発生させる。</p>
<p>プログラム例：乱数を 5 個発生させる。</p> <pre> 10 FOR I=1 TO 5 20 LPRINT RND(1) 30 NEXT I 40 END </pre> <p>実行例</p> <pre> .763171 .981958 .803397 .793264 .154362 </pre> <p>参照：RANDOMIZE</p>	<p>PC8 ・本文参照。</p> <p>TRS_{II} ・TRS_I と同じ。</p> <p>PC3 ・0 以上 A 未満の 12 桁の乱数を発生する。A が符号を伴わない数値定数、数値変数の場合や関数を伴う場合、カッコで囲む必要はない。</p> <p>IF8₂ ・CPM と同じ。</p> <p>レベル₃ ・CPM と同じ。</p> <p>C180 ・M223 と同じ。</p> <p>M243 ・M223 と同じ。</p> <p>MZB ・MZK と同じ。</p> <p>BUB ・CPM と同じ。</p>	<p>MZ35 ・PC3 と同じ。</p> <p>X1Hu ・引数は省略できる。</p> <p>MB16 ・CPM と同じ。ただし、乱数は 0 以上 1 未満である。</p> <p>IBM55 ・CPM と同じ。</p> <p>PC100 ・CPM と同じ。</p> <p>MSX ・本文と同じ。</p>

乱数の初期設定 RANDOMIZE

ランダマイズ

形式: RANDOMIZE

機能: 乱数の系列を変え、疑似乱数を予知できないものにする。

特徴: 1 シミュレーション、ゲームなどで RND を使う場合、この命令を併用すれば、乱数の信頼性があがる。
 2 RND 関数をプログラム中で使用する場合、初期設定のときに使うと便利である。

プログラム例: RANDOMIZE 文を実行した後、乱数を 10 個発生させる。

```

10 OPEN "POUT" FOR OUTPUT AS FILE 1 MODE 3
20 RANDOMIZE
30 FOR I = 1 TO 5
40 PRINT #1, RND, RND
50 NEXT I
60 CLOSE 1
70 END

```

実行例

0.883404	0.049004
0.702057	0.129996
0.820042	0.108947
0.631872	0.169425
0.688332	0.963796

参照: RND

APL —

TRS_I RANDOM

CPM RANDOMIZE A

- ・ $0 \leq A \leq 65529$
- ・ A を省略すると値を聞いてくる。
- ・ A の値によって異なる系列をつくる。

M223 ・本文参照。

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ ・BUB と同じ。レベル₃ RANDOMIZE (A)

- ・ $-32768 \leq A \leq 32767$ 。
- ・ ほかは CPM と同じ。

C180 ・本文と同じ。

M243 ・本文と同じ。

MZB —

BUB RANDOMIZE A

- ・ $-32768 \leq A \leq 32767$ を満たさなければならぬ。
- ・ ほかは CPM と同じ。

FM8 ・BUB と同じ。

PSP ・本文と同じ。

PC88 ・BUB と同じ。

N52 —

PC6 —

MLT ・BUB と同じ。ただし、式を省略してもよい。省略した場合、BASIC では "Random Number Seed (-32768 to 32767) ?" と表示して入力待ちになる。

HC ・BUB と同じ。

FP11 ・本文と同じ。

SMC ・BUB と同じ。ただし、A の省略も可能である。

MZ35 ・TRS_I と同じ。X1H_u RANDOMIZE A

- ・ $-32768 \leq A \leq 32767$ 。
- ・ 省略すると BASIC が乱数系列を変える。

MB16 ・レベル₃ と同じ。

IBM55 ・MLT と同じ。

PC100 ・BUB と同じ。

MSX —

階乗 FAC

ファクトリアル

形式：FAC (A)

機能：A の階乗を求める。

特徴：1 A は 0 ~ 33 の整数。
 2 $FAC(7) = 1 * 2 * 3 * 4 * 5 * 6 * 7$ で代用できる。

プログラム例：0~3 までの階乗を求める。

```
10 FOR I=0 TO 3
20 PRINT I,FAC(I)
30 NEXT
```

40 END

OK

実行例

RUN

0 1

1 1

2 2

3 6

OK

参照：

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 —

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu ・ 本文参照。

MB16 —

IBM55 —

PC100 —

MSX —

2進数の左/右シフト SL%, SR%

シフト・レフト%, ライト%

形式：SL% (A%, A1), SR% (A%, A1)

機能：A%をA1ビット¹⁾分、左または右へシフト²⁾する。

- 特徴：
- 1 右側または左側のA1ビット分は0になる。
 - 2 A%は2進数値³⁾であること。
 - 3 A1が負のときは、逆方向へシフトされる。
 - 4 BASIC プログラムで、アセンブラのようなビット操作をさせる時に便利である。
 - 5 DEF FNP(C)=C+65536*(C>32767)
DEF FNQ(C)=C-65536*(C<0)
としたとき SL%=FNQ(A%)/B,
SR%=FNP(FNQ(A%)*B-INT(FNQ(A%)*B/65536))
*65536 で代用できる。ただし、シフト数1, 2, 3, ...のとき
B=2, 4, 8, ...。

用語：

¹⁾ビット 情報量の基本単位で0と1の1組のこと。Binary-digitの略。
²⁾シフト 1バイトないし、2バイトの情報を、各ビットの状態を変えずに、その情報を右ないし左へ1ビット分だけスライドすること。
³⁾2進数値 4ケタまでの16進数字 (C180)。

プログラム例：2進数 "0123" を4ビット左、右へシフトする。

```
10 A1%=SL%('0123',4)
20 A2%=SL%('0123',-4)
30 PRINT A1%
40 PRINT A2%
50 END
```

実行例

```
1230
0012
```

参照：

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 —

IF8₂ —

レベル₃ —

C180 ・ 本文参照。

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 —

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

文字列の長さ LEN

レングス

形式: LEN (E\$)

機能: 文字列 E\$ の長さを求める関数。

特徴: 1 空白やプリントされない文字も数える。
2 0 から 255 の値が得られる。

プログラム例: いろいろな文字列の長さを求める。

```

10 E$="ABCDEFGH"
20 FOR I=1 TO 7
30 F$=LEFT$(E$,I)
40 A=LEN(F$)
50 LPRINT "F$=";F$,"LEN(F$)=";A
60 NEXT I
70 END

```

実行例

F\$=A	LEN(F\$)= 1
F\$=AB	LEN(F\$)= 2
F\$=ABC	LEN(F\$)= 3
F\$=ABCD	LEN(F\$)= 4
F\$=ABCDE	LEN(F\$)= 5
F\$=ABCDEF	LEN(F\$)= 6
F\$=ABCDEFGH	LEN(F\$)= 7

参照:

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ・ 本文と同じ。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。ただし、文字列の長さが 0 であってはならない。

PC3 ・ 本文と同じ。ただし、E\$ が文字定数の場合、カッコで囲む必要はない。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。

M243 ・ 本文と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FPI1 ・ 本文と同じ。

SMC ・ 本文と同じ。

MZ35 ・ 本文と同じ。ただし、E\$ が文字変数の場合、カッコではなく、クォーテーションでかこむ。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 文字列の総バイト数が得られる。
・ 2 バイト・キャラクタを含む文字列 E\$ の文字数を知るために、KLEN (E\$) がある。

PC100 ・ IBM55 と同じ。

MSX ・ 本文と同じ。

右側文字列の取り出し RIGHT\$

ライト\$

形式：RIGHT\$(E\$, A)

機能：文字列 E\$ の右から A 文字目までを求める関数。

- 特徴：
- 1 $0 \leq A < 256$ を満たしていなければならない。
 - 2 A が浮動小数点型の数値であれば、小数点以下は切り捨てられ整数値として扱われる。
 - 3 A の値が E\$ の長さより大きい場合は、E\$ 全体の長さに、0 の場合はナル・ストリングとなる。

プログラム例：文字列の右側から、1 文字ずつ増やしなが取り出す。

```
10 E$="ABCDE"
20 FOR I=1 TO 5
30 F$=RIGHT$(E$, I)
40 LPRINT "RIGHT$(E$, "; I; ")="; F$
50 NEXT I
60 END
```

実行例

```
RIGHT$(E$, 1)=E
RIGHT$(E$, 2)=DE
RIGHT$(E$, 3)=CDE
RIGHT$(E$, 4)=BCDE
RIGHT$(E$, 5)=ABCDE
```

参照：MID\$, LEFT\$

APL ・ 本文と同じ。

TRS_I ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 RIGHT(E\$, A)
・ 文字列の左からの A 桁目から最後までの文字列を求める。A の意味が本文と異なるので注意。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 本文と同じ。

IF8₂ ・ A の小数部分は四捨五入される。

レベル₃ ・ 本文と同じ。

C180 E\$(A: 508)
・ 文字変数 E\$ に対し、E\$(n:m) と書くと、E\$ の n 文字目から m 文字目の文字列を表すことになる。m として許される最大文字数をとれば RIGHT\$ と同じ働きをする。

M243 ・ M223 と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ IF8₂ と同じ。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。

SMC ・ IF8₂ と同じ。

MZ35 ・ 本文と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ IF8₂ と同じ。

IBM55 ・ IF8₂ と同じ。

PC100 ・ IF8₂ と同じ。ただし、2 バイト・キャラクタを含む場合は注意を要する。

MSX ・ 本文と同じ。

左側文字列の取り出し LEFT\$

レフト\$

形式: LEFT\$(E\$, A)

機能: 文字列 E\$ の左から A 文字目までを求める関数。

- 特徴: 1 $0 \leq A < 256$ を満たしていなければならない。
 2 A が浮動小数点型の数値であれば、小数点以下は切り捨てられ整数値として扱われる。
 3 A の値が E\$ の長さより大きい場合は、E\$ 全体の長さに、0 の場合はナル・ストリングになる。

プログラム例: 文字列の左側から 1 文字ずつ増やしなが取り出す。

```
10 E$="ABCDE"
20 FOR I=1 TO 5
30 F$=LEFT$(E$, I)
40 LPRINT "LEFT$(E$, "; I; ")="; F$
50 NEXT I
60 END
```

実行例

```
LEFT$(E$, 1 )=A
LEFT$(E$, 2 )=AB
LEFT$(E$, 3 )=ABC
LEFT$(E$, 4 )=ABCD
LEFT$(E$, 5 )=ABCDE
```

参照: MID\$, RIGHT\$

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 LEFT(E\$, A)
・ 機能は本文と同じ。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS₁₁ ・ 本文と同じ。

PC3 ・ 本文と同じ。

IF8₂ ・ 本文と同じ。ただし、A
の小数部は四捨五入される。レベル₃ ・ 本文と同じ。C180 E\$(1:A)
・ 文字変数 E\$ に対し、E\$(n:m)
と書くと、E\$ の n 文字目から m 字
目の文字列を表すことになる。
(部分文字列修飾)。n として 1 を
とれば、LEFT\$ と同じ働きをす
る。

M243 ・ M223 と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。ただし、 $0 \leq A \leq 255$ で、
小数点以下は四捨五入する。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。

SMC ・ 本文と同じ。ただし、 $0 \leq A \leq 255$ で、
小数点以下は四捨五入する。

MZ35 ・ 本文と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ IF8₂ と同じ。IBM55 ・ IF8₂ と同じ。PC100 ・ IF8₂ と同じ。ただし、2
バイト・キャラクタを含む場合は
注意を要する。

MSX ・ 本文と同じ。

中間文字列の取り出し MID\$

ミッド\$

形式：① MID\$ (E\$, A1, A2)

② MID\$ (E1\$, A1, A2) = E2\$

機能：①文字列E\$の始めからA1番目の文字に始まって、A2の長さの文字列を求める。

②E1\$において、A1文字目からA2文字を、E2\$の始めからA2文字目までの文字列で置き換える。

- 特徴：1 A1, A2は、0以上256未満の値なら浮動小数点型の値でもかまわない。ただし、小数点以下は切り捨てられる。
2 機能①において、A2が省略されていたり、A1番目の文字から右にA2文字なかったとしたら、結果はA1番目の文字から右側の全文字となる。
3 E\$の文字数がA1より小さい場合、ナル・ストリングとなる。

プログラム例：E\$からの文字列取り出しと、E1\$のE2\$により置き換える。

```
10 E$="ABCDEF"
20 E1$="ABCDEF"
30 E2$="123456"
40 MID$(E1$, 3, 2)=E2$
50 LPRINT "MID$(E$, 3, 2)="; MID$(E$, 3, 2),
  "E1$="; E1$
60 END
```

実行例

MID\$(E\$, 3, 2)=CD

E1\$=AB12EF

参照：LEFT\$, RIGHT\$

APL ・本文の機能1のみ。

TRS₁ ・本文と同じ。

CPM ・APLと同じ。

M223 MID (E\$, A1, A2)
・機能はAPLと同じ。

MZK ・APLと同じ。

CBM ・APLと同じ。

PC8 ・本文参照。

TRS_{II} ・APLと同じ。

PC3 ・APLと同じ。

IF8₂ ・APLと同じ。ただし、A
の小数部は四捨五入される。

レベル₃ ・APLと同じ。

C180 E\$ (A1: A1+A2-1)
・文字変数E\$に対し、E\$ (n: m)
と書くと、E\$のn文字目からm字
目の文字列を表すことになる。
・機能はAPLと同じ。
・配列の場合は、
F\$ (12) (A1: A1+A2-1)

M243 ・M223と同じ。

MZB ・APLと同じ。

BUB ・APLと同じ。

FM8 ・②でA2は省略でき、そ
のとき左辺の長さ分が代入される。

PSP ・APLと同じ。

PC88 ・APLと同じ。

N52 ・A1, A2は1以上256未
満。

PC6 ・APLと同じ。

MLT ・1 ≤ A1 ≤ 255, 0 ≤ A2 ≤
255で、小数点以下は四捨五入する。

HC ・特徴3はない。

FP11 ・MLTと同じ。ただし、
小数点以下は切り捨てられる。

SMC ・APLと同じ。
・②に関連して文字列中の一部を他
の文字列で置き換えるREPLACE
\$関数がある。

MZ35 ・APLと同じ。

X1Hu ・本文と同じ。

MB16 ・IF8₂と同じ。

IBM55 ・IF8₂と同じ。

PC100 ・Aの小数部は四捨五入さ
れる。
・その他は本文と同じ。

MSX ・本文と同じ。

1 文字の繰り返し STRING\$

ストリング\$

形式: STRING\$ (M1, M2)

機能: M2 のアスキー・コードで示される文字を, M1 個の文字列で与える関数.

特徴: 1 M2 に, 文字列も許されるが, 最初の文字しか有効でない.
 2 M1 は $0 \leq M1 < 256$ の範囲で, 小数点以下は切り捨てて使われる.
 3 グラフや図表などを作るのに便利である.

プログラム例: 文字列とアスキー・コードを与えて, (A) を繰り返す文字列を印字する.

```
10 A$="ABCD"
20 LPRINT STRING$(5,A$)
30 LPRINT STRING$(6,&H41)
40 END
```

実行例

```
AAAAA
AAAAAA
```

参照: STR\$

APL —

TRS_I ・本文と同じ.

CPM ・本文と同じ.

M223 ・本文と同じ.

MZK —

CBM —

PC8 ・本文参照.

TRS_{II} ・本文と同じ.

PC3 ・本文と同じ. ただし, M2 は文字定数(最初の1文字が有効).

IF8₂ ・本文と同じ.レベル₃ ・本文と同じ.

C180 ・該当なし. ただし, REP\$ (E\$, A) は E\$ を A 個つなげて文字列をつくる関数.

M243 ・本文と同じ.

MZB STRING\$ ("*", M)
・M個の連続した*を与える.

BUB ・本文と同じ.

FM8 ・本文と同じ.

PSP ・本文と同じ.

PC88 ・本文と同じ.

N52 ・M2 は JIS コードである.

PC6 —

MLT ・M1, M2 が整数式の場合,

0 ~ 255 の範囲で, 小数点以下は四捨五入する.

HC ・本文と同じ.

FP11 ・本文と同じ.

SMC ・M2 は文字列のみ. 文字列全体が繰り返される.

MZ35 ・PC3 と同じ.

X1Hu ・M2 は文字式でもよい.

MB16 ・MLT と同じ.

IBM55 ・MLT と同じ.

PC100 ・本文と同じ.

MSX ・本文と同じ.

空白の印字 SPC

スペース

形式：SPC (10)

機能：指定された数の空白の列を示す。

- 特徴：
- 1 引数は、0 から 255 までの範囲。
 - 2 SPC 関数は、PRINT 文および LPRINT 文の中でのみ(*)使える。その間は、指定された数の空白を印字する。
 - 3 引数の小数部分は、切り捨てられる。
 - 4 SPACE\$ (10) でも同じことができる。この場合代入も可。
 - 5 空白を印字するとき、SPACE\$ より簡潔に書けるので便利である。

(*)PRINT #n 文中でも使える。

プログラム例：LPRINT と組み合わせた例。

```

10 LPRINT "ABCDEF";SPC(5);"12345"
20 LPRINT SPC(5);"12345"
30 LPRINT SPC(5),"12345"
40 LPRINT SPC(10);"12345"
50 END

```

実行例

```

      ABCDEF      12345
          12345
              12345
                  12345

```

参照：PRINT, LPRINT

APL ・本文と同じ。

TRS_I ・該当なし。ただし、関数 STRING\$ (10, " ") で同じことができる。

CPM ・本文と同じ。

M223 ・SPACE\$ (10) のみ。

MZK ・本文と同じ。ただし、特徴 2 で PRINT 文および PRINT/P 文中のみで使える。

CBM ・本文と同じ。ただし、特徴 2 で PRINT 文のみで使える。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。PC3 SPA (10)
・特徴 2 で代入も可。IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。ただし、特徴 2 で PRINT 文および PRINT# 文中のみで使える。

C180 ・該当なし。ただし、関数 REP\$ (" ", 10) で同じことができる。

M243 ・M223 と同じ。

MZB ・MZK と同じ。

BUB ・本文と同じ。

FM8 ・レベル₃ と同じ。PSP ・レベル₃ と同じ。

PC88 ・本文と同じ。ただし、

PRINT #n 文中でも使えることをマニュアルに明記している。

N52 ・本文と同じ。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・SPACE\$ 関数はない。

SMC ・M223 と同じ。

MZ35 ・PC3 と同じ。

X1Hu ・本文と同じ。

MB16 ・引数の値が出力デバイスの定められた幅以上では、引数を幅で割った余りの数のスペースを出力する。

IBM55 ・MB16 と同じ。

PC100 ・本文と同じ。ただし、引数は -32768~32767 まで使えるが、負の数はすべて 0 とみなされる。

MSX ・本文と同じ。

スtringの探索 INSTR

インString

<p>形式：INSTR (M, E1\$, E2\$)</p> <p style="text-align: center;">┌ └</p> <p style="text-align: center;">省略可</p> <p>機能：文字列 E1\$ 中の M 個目から右へ文字列 E2\$ を探索し、最初に見つかった位置を与える関数。</p> <p>特徴：1 M は算術式も可能で、探索を始める位置を指定し、小数点以下は切り捨てられる。 $0 \leq M < 256$。 2 M が E1\$ の文字列数より大きい場合、E1\$ がナル・String の場合または E2\$ が見つからない場合は、関数の値は 0 となる。 3 E2\$ がナル・String の場合は、関数の値は 1 または M となる。 4 E2\$ および E1\$ は文字変数、文字式または文字列であっても可能である。 5 同じ文字列を探すのに、便利である。</p>	<p>APL —</p> <p>TRS_I —</p> <p>CPM ・本文と同じ。</p> <p>M223 ・本文と同じ。</p> <p>MZK —</p> <p>CBM —</p> <p>PC8 ・本文参照。</p> <p>TRS_{II} ・本文と同じ。</p> <p>PC3 —</p> <p>IF8₂ ・本文と同じ。</p> <p>レベル₃ ・本文と同じ。</p> <p>C180 POS (E1\$, E2\$, M) ・機能は本文と同じ。</p> <p>M243 ・本文と同じ。</p> <p>MZB —</p> <p>BUB ・本文と同じ。</p> <p>FM8 ・本文と同じ。</p> <p>PSP ・C180 と同じ。</p> <p>PC88 ・本文と同じ。さらに、配列要素中の特定値を検出する SEARCH 命令がある。</p> <p>N52 ・ $0 < M < 256$。</p> <p>PC6 —</p> <p>MLT ・N52 と同じ。</p>	<p>HC ・本文と同じ。</p> <p>FP11 ・ $1 \leq M \leq 256$。また、特徴 3 は 0 となる。</p> <p>SMC ・ M は小数点以下四捨五入。 ・ E2\$ がナル・String の場合は、関数の値は 0 または M。</p> <p>MZ35 SEARCH E1\$, M, E2\$, N ・ N には見つかった位置が入る。</p> <p>X1Hu ・ $1 \leq M \leq 256$。</p> <p>MB16 ・小数点以下は四捨五入される。</p> <p>IBM55 ・MB16 と同じ。 ・この他に、文字列 E1\$ の n 番目の文字の位置をバイト数で返す KPOS (E1\$, n) がある。</p> <p>PC100 ・IBM55 と同じ。</p> <p>MSX ・本文と同じ。</p>
<p>プログラム例：A\$ の文字列の中から "SH" の位置を探す。</p> <pre> 10 A\$="SHINSHU UNIV." 20 I=INSTR(A\$,"SH") 30 J=INSTR(3,A\$,"SH") 40 LPRINT "I=";I,"J=";J 50 END </pre> <p style="text-align: center;">実行例</p> <p style="text-align: center;">I= 1 J= 5</p>		
<p>参照：</p>		

印字桁の指定 TAB

タブ

形式：PRINT TAB (100) ; “ タテ ”

引数

機能：指定した位置まで空白をプリントする。

- 特徴：
- 1 TABはPRINTまたはLPRINT文中でしか使えない。
 - 2 もし現在のプリント位置が指定した位置を超えていれば、TABは無効となる。
 - 3 指定する位置は0から255までの範囲。
 - 4 位置0が左端である。
 - 5 表を出力する時の頭をそろえるのに利用すると便利である。

プログラム例：LPRINT文（プリンタへの打ち出し）と組み合わせた例。

```
10 A=10
20 LPRINT A
30 LPRINT TAB(10);A
40 END
```

実行例

10

10

参照：PRINT, LPRINT, PRINT #n, SPC

- APL ・本文と同じ、ただし、引数は0から255。
 ・TAB(0)は256の位置へカーソルを移す。
 ・指定した位置が現在のカーソル位置より小さければ、TABは無効となる。
 ・PRINT文中でのみ使用可。

- TRS₁ ・本文と同じ、ただし、63よりも大きい引数を指定したときは、その数から64を引いた位置まで空白を印字する。
 ・TAB関数の後にはコンマもセミコロンも必要ない。

- CPM ・本文と同じ。

- M223 ・本文と同じ、ただし、引数はCRTで最大79、プリンタで最大131。
 ・PRINT文、PRINT #n文中で用いられる。

- MZK ・本文と同じ、ただし、PRINT文中でのみ使用可。

- CBM ・MZKと同じ。

- PC8 ・本文参照。

- TRS_{II} ・指定された位置が80より大きいならば次の行に続く。
 ・TAB関数の後にはコンマもセミコロンも必要ない。

- PC3 TAB 10
 ・PRINT文中で用いる。

- IF8₂ ・レベル₃と同じ、ただし、位置は0から255まで指定でき、PRINT文、LPRINT文中でのみ使用可。

- レベル₃ ・本文と同じ、ただし、

- PRINT文中でのみ用いる。
 ・引数は0以上の数であればよい。
 ・引数が1桁の表示文字数を超えたら、1桁の表示文字数で割った余りを用いる。その値が現在のカーソル位置より小さければ、次の行で同様のことを行う。

- C180 ・本文と同じ、ただし、引数は1から80。
 ・PRINT文中でのみ使用可。

- M243 ・M223と同じ。

- MZB ・MZKと同じ。

- BUB ・本文と同じ、ただし、引数の上限は不明。

- FM8 ・本文と同じ、ただし、引数は0から32767。そのとき、1行の桁数で割った余りの数が桁位置として指定される。
 ・PRINT文、PRINT #n文中で用いられる。

- PSP TAB (桁, 行)
 ・機能は本文と同じ、そのなかに行も指定できる。
 ・桁の上限は79または35。
 ・行の上限は25または24で省略可、CRTのみ有効。

- PC88 ・本文と同じ。

- N52 ・本文と同じ。

- PC6 ・PRINT #1でも使える。

- MLT ・引数は-32768～32767で、負か0はTAB(1)となり、80以上は80で割った余りが入る。

- HC ・本文と同じ。

印字桁の指定 TAB

タブ

<p>FPI1 ・指定位置が現在のカーソル位置より小さければ、次の行へ改行し、次の行の先頭から数えた位置が指定される。</p> <p>・一行の範囲を超えた指定を受けたとき、現在の行の先頭から数えた位置が指定される。</p>			
<p>SMC ・プリント位置が指定位置を超えていれば、次の行の TAB 位置に印字する。</p>			
<p>MZ35 ・PC3 と同じ。</p>			
<p>X1Hu ・本文と同じ。</p>			
<p>MB16 ・引数は小数点以下は四捨五入される。引数が0であれば1と同じ。</p> <p>・引数が現在カーソル位置より小さい指定をした場合は、次の行の指定位置に移動する。</p> <p>・1行の表示文字数を越える場合は、引数を表示文字数で割った余りに指定される。</p>			
<p>IBM55 ・MB16と同じ。また、引数の負の数は0とみなされる。</p> <p>・PRINT# でも使える。</p>			
<p>PC100 ・引数は -32768 ~ 32767 の範囲。ただし、負の数は0とみなされる。</p>			
<p>MSX ・本文と同じ。</p>			

時刻 TIME\$

タイム\$

形式: TIME\$

機能: 内蔵クロックの時刻を示す。

- 特徴: 1 TIME\$は内蔵クロックの時刻を時分秒 HH:MM:SS の形式で示す。なお、HHは0~23、MMとSSは0~59の範囲。
- 2 内蔵クロックは、TIME\$="11:22:33"の形式の文字変数として設定できる。TIME\$に値を代入することもできる。ただし、INPUT文で直接TIME\$に代入することはできない。
- 3 電源投入時には、内蔵クロックは00:00:00に設定される。
- 4 内蔵クロックをあらかじめ設定しないときは、TIME\$は使用時間を示す。
- 5 時間によって実行を制御するのに便利である。

プログラム例: 10秒ごとに時刻を表示する例。

```
10 TIME$="00:00:00"
20 IF RIGHT$(TIME$,1)<>"0" THEN GOTO 20
30 PRINT TIME$
40 FOR I=1 TO 1000:NEXT I
50 GOTO 20
```

実行例

```
00:00:00
00:00:10
00:00:20
00:00:30
00:00:40
00:00:50
```

参照: DATE\$, SLEEP, WAIT

APL —

TRS₁ ・日付けと時刻をMM/DD/YY HH:MM:SSの形式で示す。
・日付けと時刻の設定はDOSモードで行う。

CPM —

M223 TIME
・電源投入時からの時間を秒単位の整数値で示す。

MZK TI\$
・機能は本文と同じ。ただし、TI\$の形式はHHMMSS。

CBM ・MZKと同じ。

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。ただし、TIME\$の形式はHH、MM、SS。

PC3 —

IF8₂ ・M223と同じ。レベル₃ ・M223と同じ。

C180 ・M223と同じ。

M243 ・M223と同じ。

MZB ・MZKと同じ。

BUB ・M223と同じ。

FM8 ・本文と同じ。
・TIMEは積算時間関数。

PSP TIM
・機能はM223と同じ。

PC88 ・現在の時刻を示す。

N52 ・PC88と同じ。

PC6 TIME
・1/512秒ごとに2つつ増加するタイマ。
・ハードの影響を受けるため、正確なタイマとしては使用不可。

MLT ・M223と同じ。
・設定はTIME=M1で行う。

HC ・特徴3、4はない。

FPI1 —

SMC TIME\$(N)
・0≤N≤4でなければならない。
N=0: HHMMSS
N=1: HH:MM:SS
N=2: HH:MM:SS
ただし、12時間方式
N=3: "AM" または、"PM"を返す。
N=4: 午前の時"0", 午後の時"1"を返す。
・Nを省略すると0を指定したことになる。
・設定にはSET TIMEを用いる。

MZ35 TIMEの形式は、整数部が月日時(24時間制)を、小数部が分秒(60進)。

X1Hu ・PC88と同じ。また、秒数だけのタイマ、TIMEがある。

MB16 ・TIMEは0時0分から現在の時刻までの秒数。

IBM55 ・PC88と同じ。なお、設定時、秒は常に"00"となる。

PC100 ・PC88と同じ。さらに、0時からの積算時間を与えるTIMER関数がある。

時刻 TIME\$

タイム\$

<p>MSX TIME</p> <ul style="list-style-type: none"> ・電源投入時からの時間を1/60秒単位の整数値で表す。 ・設定は TIME=M1 で行う。 			
---	--	--	--

日付け DATE\$

デイト\$

形式: DATE\$

機能: 内蔵されたクロック¹⁾の示す日付けを表す。

- 特徴:
- 1 プログラムから日付けを読み出すために使用する。
 - 2 日付けの形式は、YY/MM/DD。YYが年、MMが月、DDが日を表し、それぞれ2桁の数字列である。
 - 3 電源投入時には、79/01/01にセットされている。
 - 4 DATE\$には新しい値を代入することもできる。
 - 5 日付けを設定したり、表示したいときに使うと便利である。

用語: ¹⁾クロック 内蔵タイマのこと。一定の時間間隔で、パルスを出していて、その間隔は機種によって異なる。

プログラム例: 日付けを印刷し、次に日付けを変更してまた印刷する。

```
10 LPRINT DATE$
20 DATE$="82/01/10"
30 LPRINT DATE$
40 END
```

実行例

```
79/01/01
82/01/10
```

参照: TIME\$

APL —

TRS₁ TIME\$

- ・日付けの形式は、MM/DD/YY
- HH:MM:SS、(月/日/年 時:分:秒)である。

CPM —

M223 DATE

- ・日付けの形式は、DD-MM-YY (日-月-年)である。

MZK —

CBM —

PC8 ・本文参照。

TRS₂ ・本文と同じ。ただし、日付けの形式は、
WWWMMDDYYYYNNN_,ST
〔曜日(英文字)、月(英文字)、日(数字)、西暦年(数字)、元旦からN日、S番目の月で週のT日目〕。

PC3 DISP TIME

- ・CRT上に時刻を表示する。整数部が、月・日と時(24時間制)、小数部が分秒(60進法)を表す。

IF8₂ DATE\$(M)

- ・Mを省略した場合は本文と同じ。
- Mは1月1日からの通算日数とみなし、日付けに変換する。

レベル₃ ・本文と同じ。ただし、うるう年は西暦2000年までは自動的に計算される。

C180 ・本文と同じ。ただし、24時を越えても、年月日の更新はしない。

M243 ・M223と同じ。

MZB —

BUB ・IF8₂と同じ。

FM8 ・本文と同じ。

- ・電源投入時はすべてゼロ。

PSP —

PC88 ・本文と同じ。ただし、日付けは更新され、正しい日付けとなっている。

N52 ・PC88と同じ。

PC6 —

MLT ・電源投入時は、1978—01—01となっている。

- ・日付けの形式は、YY-MM-DD、YY/MM/DD、YYYY-MM-DD、YYYY/MM/DD、(年-月-日 or 年/月/日)である。
- 年の指定を2桁で行った場合、西暦2000~2099年の指定をしたことになる。

HC ・本文と同じ。

FP11 —

SMC DATE\$(N)

- ・Nは0≤N≤2である数値、変数、式であること。
- N=0: MMDDYY
- N=1: MM/DD/YY
- N=2: YY/MM/DD (YYが年、MMが月、DDが日)
- ・設定はSET DATEを用いる。
- ・機能はPC88と同じ。
- ・これに関連してDATE Mとすると、Mには1月1日からの経過日数が入る。

MZ35 ・PC3と同じ。

日付け DATE\$

デイト\$

<p>X1Hu ・年は更新されない。</p> <hr/> <p>MB16 ・日付けは更新される。日付け型式は MLT と同じ。 ・年の範囲は1980～2077まで。 ・DATE はその年の 1月1 日から。</p> <hr/> <p>IBM55 ・PC88 と同じ。 ・型式は MM-DD-YY である。</p> <hr/> <p>PC100 ・型式は DD-MM-YYYY である。 ・日付けは更新されて正しい日付けとなっている。</p> <hr/> <p>MSX —</p> <hr/>			
--	--	--	--

曜日 DAY

デイ

形式：DAY

機能：現在の曜日に対応する値（1から7）が入っている。

特徴：1 1は（日）で、7は（土）である。
 2 一度正しい曜日を入力すると、セットしなおす必要はない。
 3
$$\text{DEF FNDAY}(Y, M, D) = (((26 * ((M - 2) - 12 * (M <= 2)) - 2) * Y + 10) + D + ((Y + (M <= 2)) \text{MOD } 100) + ((Y + (M <= 2)) \text{MOD } 100) + (((Y + (M <= 2)) * Y + 100) * 4) - 2 * ((Y + M <= 2) * Y + 100) \text{MOD } 7) + 1$$
 としたとき、
 FNDAY (1984, 4, 1)
 とすると対応する曜日が1～7で与えられ DAY の代用になる。

プログラム例：DAY関数を用いて、曜日をカタカナで表示する。

```
10 REM *** DAY TEST PROG
20 ON DAY GOTO 30,40,50,
60,70,80,90
30 LPRINT "ニャウヒ" : GOTO
100
40 LPRINT "カツヨウヒ" : GOTO
100
50 LPRINT "カヨウヒ" : GOTO 1
60 LPRINT "スイヨウヒ" : GOTO
100
70 LPRINT "モクヨウヒ" : GOTO
100
80 LPRINT "キンヨウヒ" : GOTO
100
90 LPRINT "トヨウヒ" : GOTO
100
100 LPRINT DAY
110 END
```

実行例

```
RUN
モクヨウヒ
5
```

参照：DATE\$, TIME\$

APL —

TRS₁ —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS₁₁ —

PC3 —

IF8₂ ・BUBと同じ。レベル₃ —

C180 —

M243 —

MZB —

BUB ・DAYはSMCのDAY\$
 (1)と、DAY\$はSMCのDAY
 \$ (0)と同じ。

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC ・本文参照。

FP11 —

SMC DAY\$ (N)
 ・Nが1の時は1文字〔0 (日) から6 (土)〕が与えられる。
 ・Nが0が省略された時は3文字
 (“SUN” から “SAT”) が与えられる。

MZ35 —

X1Hu DAY\$
 ・“SUN” から “SAT” が与えられる。

MB16 —

IBM55 —

PC100 —

MSX —

整数化 INT

インテジャ

<p>形式：INT (A)</p> <p>機能：A の値を超えない最大の整数を与える関数。</p> <p>特徴：</p> <ol style="list-style-type: none"> 1 CINT との違いは、与えられる値が浮動小数点型であるということ、負数における値が異なることである。 2 FIX との違いは、A が負のとき FIX は A より 1 だけ大きな数を与えるという点である。すなわち、 $\text{INT}(-14.5) = -15$ $\text{CINT}(-14.5) = -14$ $\text{FIX}(-14.5) = -14$ 	<p>APL ・ 本文と同じ。</p> <p>TRS_I ・ 本文と同じ。</p> <p>CPM ・ 本文と同じ。</p> <p>M223 ・ 本文と同じ。ただし、 INT # (C#) で高精度演算もできる。</p> <p>MZK ・ 本文と同じ。</p> <p>CBM ・ 本文と同じ。</p> <p>PC8 ・ 本文参照。</p> <p>TRS_{II} ・ 本文と同じ。</p> <p>PC3 ・ 本文と同じ。ただし、A が負符号を伴わない数値定数、数値変数の場合や関数を伴う場合、カッコで囲む必要はない。</p>	<p>MLT ・ 本文と同じ。</p> <p>HC ・ 本文と同じ。</p> <p>FP11 ・ 本文と同じ。さらに、引数の型により倍々精度まで求められる。 ・これに関連して、小数部分を与える関数 FRAC もある。</p> <p>SMC ・ 本文と同じ。</p> <p>MZ35 ・ PC3 と同じ。</p> <p>X1Hu ・ 本文と同じ。</p> <p>MB16 ・ 本文と同じ。</p> <p>IBM55 ・ 本文と同じ。</p> <p>PC100 ・ 本文と同じ。</p> <p>MSX ・ 本文と同じ。</p>
<p>プログラム例：A1, A2 の値を超え、最大の整数を、それぞれ A3, A4 へ代入する。ただし、A3, A4 は浮動小数点型の数値をとる。</p> <pre> 10 A1=123.4 : A2=-123.4 20 A3=INT(A1) : A4=INT(A2) 30 LPRINT "INT(123.4)=";A3 40 LPRINT "INT(-123.4)=";A4 50 END </pre> <p>実行例</p> <pre> INT(123.4)= 123 INT(-123.4)=-124 </pre>	<p>IF8₂ ・ 本文と同じ。</p> <p>レベル₃ ・ 本文と同じ。</p> <p>C180 ・ 本文と同じ。</p> <p>M243 ・ M223 と同じ。</p> <p>MZB ・ 本文と同じ。</p> <p>BUB ・ 本文と同じ。</p> <p>FM8 ・ 本文と同じ。</p> <p>PSP ・ 本文と同じ。</p> <p>PC88 ・ 本文と同じ。</p> <p>N52 ・ 本文と同じ。</p> <p>PC6 ・ 本文と同じ。</p>	
<p>参照：FIX, CINT</p>		

整数型化 CINT

コンバート・インテジャ

形式：CINT (A)

機能：小数部分を切り捨て、A を整数型に変換する。

- 特徴： 1 結果が整数型で与えられるので、A の値が-32768から32767までの範囲になければならない。
 2 機能は FIX 関数と同じであるが、結果が整数型になるところが FIX 関数と異なる (CINT (-14.5) は-14となる)。

プログラム例：A1 の整数部分を整数型に型変換して A2 へ代入する。

```
10 A1=123.45
20 A2=CINT (A1)
30 LPRINT "A1=";A1,"A2=";A2
40 END
```

実行例

A1= 123.45 A2= 123

参照：CDBL, CSNG, INT, FIX

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・該当なし。
 ・CEIL (A) は A より大きい数のうち、いちばん小さい整数 (上限) を求める関数。

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・N52 と同じ。

N52 ・小数点以下を四捨五入して、整数に変換する。

PC6 —

MLT ・N52 と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・N52 と同じ。また、アドレス・データを整数型に変換する CADR 関数もある。

MZ35 —

X1Hu ・N52 と同じ。

MB16 ・N52 と同じ。

IBM55 ・N52 と同じ。

PC100 ・N52 と同じ。

MSX ・本文と同じ。

フィックス

整数部分 FIX

形式：FIX (A)

機能：A の整数部分を与える関数。

特徴：1 A の値が負のとき INT と違い、FIX は A の値より大きい整数を与える。

プログラム例：A1, A2 の整数部分を A3, A4 へそれぞれ代入する。

```

10 A1=50.5 : A2=-50.5
20 A3=FIX(A1) : A4=FIX(A2)
30 PRINT "A1=";A1,"A2=";A2
40 PRINT "A3=";A3,"A4=";A4
50 END

```

実行例

```

A1= 50.5      A2=-50.5
A3= 50        A4=-50

```

参照：INT, CINT

APL ・該当なし。ただし、SGN (A)*INT (ABS (A)) で同様のことができる。

TRS_I ・本文と同じ。

CPM ・本文と同じ。

M223 ・本文と同じ。

MZK ・APL と同じ。

CBM ・APL と同じ。

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・APL と同じ。

IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。

C180 IP (A)
・機能は本文と同じ。

M243 ・本文と同じ。

MZB ・APL と同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・APL と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FPI1 ・本文と同じ。

SMC ・本文と同じ。

MZ35 ・APL と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX ・本文と同じ。

単精度の倍精度化 CDBL

コンバート・ダブル

形式：CDBL (A)

機能：単精度の数値を倍精度の数値へ変換する。

- 特徴：
- 1 2進¹⁾表現のため、増えた桁には0が入らず、誤差が生じるので注意が必要。
 - 2 C# = A のように倍精度変換へ代入することによっても倍精度化は可能である。文や式の途中で用いるときは、このCDBL関数が便利である。

用語：¹⁾2進数 0と1だけで表される数値、例えば10進数で12は、2進数で1100、□16進数

プログラム例：単精度数Aの値を倍精度数C#へ型変換して代入する。

```
10 A=1.234
20 C#=CDBL(A)
30 LPRINT "A=";A,"C#=";C#
40 END
```

実行例

A= 1.234 C#= 1.233999967575073

参照：CINT, CSNG

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 DCML (A)
・高精度数になると、内部表現が10進表現 (BCD) になるので変換で誤差は生じない。

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 —

M243 ・M223と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・変換に誤差は生じない。

SMC —

MZ35 —

X1Hu ・数値の型を倍精度にする。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。
・この他に、文字列X\$に含まれる1バイト・コード文字を2バイト・コード文字に変換する CDBL \$ (X\$) がある。

PC100 ・単精度だけでなく、整数も変換できる。
・CDBL \$ に関してはIBM55と同じ。

MSX ・本文と同じ。

倍精度の単精度化 CSNG

コンバート・シングル

<p>形式: CSNG (C#)</p> <p>機能: 倍精度の数値を単精度の数値へ変換する。</p> <p>特徴: 1 内部表現で変数のバイト数が節約され、数値演算機能、速度の面で有利になる。 2 四捨五入による丸め誤差が生じる。 3 A = C#のように単精度変数へ代入することによっても単精度化は可能である。文や式の途中で用いるときはこの CSNG 関数が便利である。</p>	<p>APL —</p> <p>TRS₁ ・本文と同じ。</p> <p>CPM ・本文と同じ。</p> <p>M223 FLOT (C#)</p> <p>MZK —</p> <p>CBM —</p> <p>PC8 ・本文参照。</p> <p>TRS₁₁ ・本文と同じ。</p> <p>PC3 —</p> <p>IF8₂ ・本文と同じ。</p> <p>レベル₃ ・本文と同じ。</p>	<p>精度数値への変換を行う CFIN 関数がある。</p> <p>SMC ・該当なし。ただし、任意の数値データを実数型に変換する CFLOAT 関数がある。</p> <p>MZ35 —</p> <p>X1Hu ・有効数字 9 桁目で四捨五入。</p> <p>MB16 ・変換した値が、$-1.70141 \times 10^{38} \sim 1.740141 \times 10^{38}$ を越えるとエラー。</p> <p>IBM55 ・本文と同じ。 ・この他に、文字列 X\$ に含まれる 2 バイト・コードの文字を 1 バイト・コードの文字に変換する CSNG\$ (X\$) がある。</p>
<p>プログラム例: 倍精度数 C# の値を単精度 A へ型変換して代入する。</p> <pre> 10 C#=1.23456789# 20 A=CSNG(C#) 30 LPRINT "C#=";C#,"A=";A 40 END </pre> <p>実行例</p> <p>C#= 1.23456789 A= 1.23457</p> <p>参照: CINT, CDBL</p>	<p>C180 —</p> <p>M243 ・M223 と同じ。</p> <p>MZB —</p> <p>BUB ・本文と同じ。</p> <p>FM8 ・本文と同じ。</p> <p>PSP ・本文と同じ。</p> <p>PC88 ・本文と同じ。</p> <p>N52 ・本文と同じ。</p> <p>PC6 —</p> <p>MLT ・本文と同じ。</p> <p>HC ・本文と同じ。</p> <p>FP11 ・本文と同じ。また、倍々</p>	<p>PC100 ・整数も変換できる。 ・CSNG\$ については IBM55 と同じ。</p> <p>MSX CSNG (C)</p>

内部コードの整数化 CVI

コンバート・インテジャ

形式：CVI (E\$)

機能：文字で表現された引数を、数値データに変換する。

- 特徴：
- 1 E\$は2バイトの文字列で、整数に変換される。
 - 2 2文字未満の場合はエラーとなり、3文字以上の場合には最初の2文字が用いられる。
 - 3 MKI\$と逆の機能を行う。
 - 4 ランダム・ファイルから読み出した内部型式の数値は、文字型から数値型に変換する必要があり、そのときに用いる。

プログラム例：“afile”から2文字ずつ読み込み、数値に変換して出力し、同時にそのときのセクタ番号とレコード番号を出力する。

```

10 OPEN "afile" AS #1
20 FIELD #1,2 AS E$
30 FOR I=1 TO 5
40 GET #1
50 A%=CVI(E$)
60 LPRINT "e$=";E$;" cvi=";A%;
70 LPRINT " fpos=";FPOS(1);
80 LPRINT " loc=";LOC(1)
90 NEXT I
100 CLOSE #1
110 END

```

実行例

e\$=ta	cvi= 24948	fpos= 81	loc= 1
e\$=ke	cvi= 25963	fpos= 82	loc= 2
e\$=ch	cvi= 26723	fpos= 83	loc= 3
e\$=an	cvi= 28257	fpos= 84	loc= 4
e\$=?!	cvi= 8511	fpos= 85	loc= 5

参照：MKI\$, MKS\$, MKD\$, CVS, CVD

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・RECORDSIZ を使えば
必要ない。

MZK —

CBM —

PC8 ・本文参照。

TRS_{II} ・本文と同じ。

PC3 ・RFORMAT# を使えば必
要ない。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・OPEN# 文で固定長レコ
ードを指定すれば必要ない。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・GET# 文での変数の形式
によるため必要ない。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・2文字未満の場合、右側に
ナールが付加される。

SMC —

MZ35 ・PC3 と同じ。

X1Hu ・本文と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX —

内部コードの単精度数値化 CVS

コンバート・シングル

形式：CVS (E\$)

機能：文字で表現された引数を、数値データに変換する。

- 特徴：1 E\$ は 4 バイトの文字列で、単精度型式の数値に変換する。
 2 4 文字未満の場合はエラーとなり、5 文字以上の場合には最初の 4 文字が用いられる。
 3 MKS\$ と逆の機能を行う。
 4 ランダム・ファイルから読み出した内部型式の数値は、文字型から数値型に変換する必要があり、そのときに用いる。

プログラム例：“afile” から 4 文字ずつ読み込み、数値に変換して出力する。

```
10 OPEN "afile" AS #1
20 FIELD #1,4 AS E$
30 FOR I=1 TO 5
40 GET #1,I
50 A=CVS(E$)
60 LPRINT "e$=";E$;" cvs=";A
70 NEXT I
80 CLOSE #1
90 END
```

実行例

```
e$=jbcd cvs= 3.30888E-09
e$=ujkl cvs= 8.76993E-07
e$= 0 cvs=-3807.87
e$=u:yu cvs= 4.20052E-04
e$=saKe cvs= 6.85048E-09
```

参照：MKI\$, MKS\$, MKD\$, CVI, CVD

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・RECORDSIZ を使えば
必要ない。

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。PC3 ・RFORMAT# を使えば必
要ない。IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。C180 ・OPEN # 文で固定長レコ
ードを指定すれば必要ない。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・GET# 文での変数の形式
によるため必要ない。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・E\$ は 5 バイトの文字列。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・E\$ は 6 バイトの文字列。
これより短い時、右側にナルが付
加され、長い時、余った文字は無
視される。

SMC —

MZ35 ・PC3 と同じ。

X1Hu ・PC6 と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX —

内部コードの倍精度数値化 CVD

コンバート・ダブル

形式: CVD (E\$)

機能: 文字で表現された引数を、数値データに変換する。

- 特徴:
- 1 E\$ は 8 バイトの文字列で、倍精度型式の数値に変換される。
 - 2 8 文字未満の場合はエラーとなり、9 文字以上の場合には最初の 8 文字が用いられる。
 - 3 MKD\$ と逆の機能を行う。
 - 4 ランダム・ファイルから読み出した内部型式の数値は、文字型から数値型に変換する必要があり、そのときに用いる。

プログラム例: "afile" から 8 文字ずつ読み込み数値に変換して出力する。

```
10 OPEN "afile" AS #1
20 FIELD #1,8 AS E$
30 FOR I=1 TO 5
40 GET #1,I
50 A#=CVD(E$)
60 LPRINT "e$=";E$;" cvd=";A#
70 NEXT I
80 CLOSE #1
90 END
```

実行例

```
e$=jbcdefgh cvd= 5.38770073608727D-08
e$=ijklmnop cvd= 1.427122099222655D-05
e$= 0 Qq/c cvd= 1.234567899876543D+16
e$=u:¥u+lg cvd= 7410.369852753195
e$=sakechan cvd= 3.358837465423164D-06
```

参照: MKI\$, MKS\$, MKD\$, CVI, CVS

APL —

TRS1 — 本文と同じ。

CPM — 本文と同じ。

M223 — RECORDSIZ を使えば
必要ない。

MZK —

CBM —

PC8 — 本文参照。

TRS11 — 本文と同じ。

PC3 — RFORMAT# を使えば必
要ない。

IF82 — 本文と同じ。

レベル3 — 本文と同じ。

C180 — OPEN # 文で固定長レコ
ードを指定すれば必要ない。

M243 — M223 と同じ。

MZB —

BUB — 本文と同じ。

FM8 — 本文と同じ。

PSP — GET# 文での変数の形式
によるため必要ない。

PC88 — 本文と同じ。

N52 — 本文と同じ。

PC6 —

MLT — 本文と同じ。

HC — 本文と同じ。

FPI1 — E\$ は 11 バイトの文字列。
これより短かい時、右側にナルが
付加され、長い時、余った文字は
無視される。

SMC —

MZ35 — PC3 と同じ。

X1Hu — 本文と同じ。

MB16 — 本文と同じ。

IBM55 — 本文と同じ。

PC100 — 本文と同じ。

MSX —

整数値の内部コード化 MKI\$

メイフ・インテジャ\$

形式：MKI\$ (M%)

機能：整数値を内部コードを表す文字列に変換する。

- 特徴：
- 1 整数を2バイトの文字列に変換する。
 - 2 小数部分は切り捨てられる。
 - 3 CVI と逆の機能を行う。
 - 4 数値をランダム・ファイルに LSET または RSET 文で書き込むときに用いる。

プログラム例：整数型のデータを読み込み、文字に変換して出力し“afile”に格納する。同時にそのときのレコード番号を出力し、最大のレコード番号も出力する。

```

10 OPEN "afile" AS #1
20 FIELD #1,2 AS E$
30 FOR I=1 TO 5
40 INPUT "data";A%
50 LSET E$=MKI$(A%)
60 LPRINT "a%=";A%;" mki=";E$;
70 PUT #1,I
80 LPRINT " loc=";LOC(1)
90 NEXT I
100 CLOSE #1
110 OPEN "afile" AS #1
120 LPRINT "lof=";LOF(1)
130 CLOSE #1
140 END

```

実行例

```

a%= 24948    mki=ta    loc= 1
a%= 25963    mki=ke    loc= 2
a%= 26723    mki=ch    loc= 3
a%= 28257    mki=an    loc= 4
a%= 8511     mki=?!    loc= 5
lof= 70

```

参照：CVI, CVS, CVD, MKS\$, MKD\$, STR\$

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・RECORDSIZ を使えば
必要ない。

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。PC3 ・RFORMAT# を使えば必
要ない。IF8₂ ・本文と同じ。レベル₃ ・本文と同じ。C180 ・OPEN # 文で固定長レコ
ードを指定すれば必要ない。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・PUT# 文での変数の形式
によるため必要ない。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 —

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC —

MZ35 ・PC3 と同じ。

X1Hu ・本文と同じ。

MB16 ・小数部分は四捨五入され
る。

IBM55 ・MB16 と同じ。

PC100 ・本文と同じ。

MSX —

単精度数値の内部コード化 MKS\$

メイク・シングル\$

形式：MKS\$ (A)

機能：単精度型式の数値を、内部コードを表す文字列に変換する。

- 特徴：
- 1 単精度数値を4バイトの文字列に変換する。
 - 2 CVS と逆の機能を行う。
 - 3 数値をランダム・ファイルに LSET または RSET 文で書き込むときに用いる。

プログラム例：単精度型式のデータを読み込み、文字に変換して出力し "afile" に格納する。

```
10 OPEN "afile" AS #1
20 FIELD #1,4 AS E$
30 FOR I=1 TO 5
40 INPUT "data";A
50 LSET E$=MKS$(A)
60 LPRINT "a=";A;" mks$=";E$
70 PUT #1,I
80 NEXT I
90 CLOSE #1
```

実行例

```
a= 123.123 mks$=>v■
a= 789456 mks$=s0~
a= 3.30888E-09 mks$=mbcd
a= 8.76992E-07 mks$=njkl
a= 6.85048E-09 mks$=sake
```

参照：CVI, CVS, CVD, MKI\$, MKD\$

APL —

TRS₁ ・本文と同じ。

CPM ・本文と同じ。

M223 ・RECORDSIZ を使えば
必要ない。

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 ・RFORMAT# を使えば必
要ない。

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 ・OPEN # 文で固定長レコ
ードを指定すれば必要ない。

M243 ・M223 と同じ。

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・PUT# 文での変数の形式
によるため必要ない。

PC88 ・本文と同じ。

N52 ・本文と同じ。

PC6 ・5バイトの文字列に変換
される。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・数値は6バイトの文字列
に変換される。

SMC —

MZ35 ・PC3 と同じ。

X1Hu ・PC6 と同じ。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。

PC100 ・本文と同じ。

MSX —

出力形式の指定 NORMAL, FIXED, FLOAT

ノーマル, フィックスド, フLOAT

形式: ① NORMAL

② FIXED 3.5

③ FLOAT 6.5

機能: ①演算結果の表示を, 標準型式で行うことを指定する.

②固定小数点型式での表示を指定する.

③浮動小数点型式での表示を指定する.

特徴: 1 ①は, 有効桁数は12桁.

②は, 上位のゼロや, 小数点以下の下位のゼロは削除される.

③は, 絶対値が1以上, 10^{12} 未満の数は, 有効桁数を現した固定小数点型式で表示する. 絶対値が 10^{-4} 未満で, 小数点以下11桁で正確にあらわされないものや, 絶対値が 10^{12} 以上のものは, 浮動小数点型式で現される. その他は, 固定小数点型式である.

2 ①は, パラメータ m, n, m と n の意味は次の通り, m: 小数点以下の桁数(0~11の整数), n: ラウンド数(0~9の整数), n=9のときは切り上げ, n=0のときは切り捨てになる. また, その他のとき(9-n)捨(10-n)入の丸めになる.

②は, 数値の絶対値が 10^{12} 以上の場合には, 浮動小数点型式になる.

3 ①は, パラメータの意味は, FIXEDの場合と同じ.

②は, $-d, d \cdots d E \pm dd$ の形で出力する. 1番目のdは0でない(ただし, 数値が0でないとき).

プログラム例: 種々の指定で印字させる. 最後に五捨六入の丸めを行っていることに注意.

```

10 NORMAL
20 A=3.14159265354
30 B=1.234E-12
40 PRINT A,B
50 FIXED 3.5
60 PRINT A,B
70 FLOAT 3.4
80 PRINT A,B
90 END

```

実行例

```

3.14159265354      1.234E-12
3.142              0.000
3.141E+00          1.234E-12

```

参照:

APL —

TRS₁ —

CPM —

M223 —

MZX —

CBM —

PC8 —

TRS_{II} —

PC3 — 本文参照.

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 — 本文と同じ.

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

アスキー・コードの文字化 CHR\$

キャラクタ\$

形式: CHR\$(A)

機能: ASCII コードがAである文字を求める関数。

- 特徴:
- 1 $0 \leq A < 256$ を満たしていなければならない。
 - 2 Aが浮動小数点型の数値であれば、小数点以下は切り捨てられ整数値として扱われる。
 - 3 特別な文字を出力する場合や、コントロール文字を出力するときを使うと便利である。
 - 4 ASCの逆である。
 - 5 ただし、コントロール文字は機種により異なるので注意が必要。

用語: ¹⁾JIS 16進コード JIS 情報交換用漢字符号系で定められているコード。
²⁾JIS 区点コード JIS 情報交換用漢字符号系の区点番号をコード化したもの。

プログラム例: ASCII コード 65~70 までの表す文字を順に表示する。

```
10 FOR I=65 TO 70
20 LPRINT "CHR$(";"I;"")="";CHR$(I)
30 NEXT I
40 END
```

実行例

```
CHR$( 65 )=A
CHR$( 66 )=B
CHR$( 67 )=C
CHR$( 68 )=D
CHR$( 69 )=E
CHR$( 70 )=F
```

参照: ASC

APL ・ 本文と同じ。

TRS_I ・ 本文と同じ。

CPM ・ 本文と同じ。ただし、カタカナは使えない。

M223 ・ 本文と同じ。ただし、ESC (27₁₀) は出力されない。

MZK ・ 本文と同じ。ただし、カタカナ・コードが特殊である、また使えないコードがある。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 本文と同じ。ただし、Aが符号を伴わない数値定数、数値変数の場合や関数を伴う場合、カッコで囲む必要はない。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。

M243 ・ M223 と同じ。

MZB ・ MZK と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ JIS コードがAである文字を求める関数。

・ ほかは本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。
 ・ さらに、倍々精度型式の数値を内部コードの文字列に変換する MKF\$ 関数もある。

SMC ・ A が整数でない場合、小数点以下は四捨五入される。

MZ35 ・ 本文と同じ。

X1Hu ・ 引数は (,) で区切られれば文字列が与えられる。
 HEXCHR\$ ("30")
 ・ 16進数で与えられた文字列を文字に変換する。
 MIRROR\$ ("A")
 ・ 与えられた文字式の 2 進数を転置したコードを与える。

MB16 ・ 小数点以下は四捨五入される。

IBM55 ・ アスキー・コードではなく、連続コードの文字化である。
 ・ $0 \leq A \leq 11535$ 。
 ・ Aに(&J), (&K) を用いることによって、JIS 16進コード¹⁾, JIS 区点コード²⁾ による文字が求められる。

PC100 ・ 数式が複数指定されたときはそのバイト数だけ連結された文字列を返す。
 ・ 数式が&H80~&HFFの範囲にあるときはキャラクタ・モードがグラフィックか漢字かによって返される文字が異なる。

アスキー・コードの文字化 CHR\$

キャラクタ\$

MSX ・本文と同じ。			

アスキー・コード化 ASC

アスキー

形式: ASC (E\$)

機能: 文字列 E\$ の最初の文字の ASCII コードの値を求める関数。

- 特徴: 1 値は10進数で得られる。
 2 文字列 E\$ の長さが0であってはならない。
 3 CHR\$ の逆である。

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122
0	1	2	3	4	5	6	7	8	9	┐	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
48	49	50	51	52	53	54	55	56	57	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47

プログラム例: A, B, C の ASCII コードを 10 進数でそれぞれ求める。

```
10 E$="ABC"
20 FOR I=1 TO 3
30 E1$=MID$(E$,I,1)
35 A=ASC(E1$)
40 LPRINT "ASC(";E1$;")=";A
50 NEXT I
60 END
```

実行例

```
ASC(A)= 65
ASC(B)= 66
ASC(C)= 67
```

参照: CHR\$

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 ASCII (E\$)

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS_{II} ・ 本文と同じ。

PC3 ・ 本文と同じ。ただし、
 ASCII コードでなく JIS コードの
 値となる。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。
 ・ ASC%(E\$) とすれば、16 進コ
 ードで結果が求まる。

M243 ・ 本文と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP ・ 本文と同じ。

PC88 ・ 本文と同じ。

N52 ・ PC3 と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。
 ・ さらに16バイトの文字列を倍々精
 度型式の数値に変換する CVF 関
 数もある。

SMC ・ 本文と同じ。

MZ35 ・ PC3 と同じ。

X1Hu ・ 本文と同じ。

MB16 ・ 本文と同じ。

IBM55 ・ 連続コードの値を求める。
 ・ この他に、JIS16進コードを求め
 る JIS\$(E\$)、JIS区点コード
 を求める KTN\$(E\$) がある。

PC100 ・ " はデータとして扱うこ
 とはできない。
 ・ その他は IBM55 と同じ。

MSX ・ 本文と同じ。

文字列の数値化 VAL

バリユー

形式：VAL (E\$)

機能：文字列 E\$ の表す10進表示の数字を数値化する関数。

- 特徴：1 E\$ の最初の文字が+、-、&、空白または数字でなければ0になる。
 2 値によって自動的に指数表示になる。
 3 数字でない文字列が混在した場合、それ以後の文字は無視される。
 4 指数表示文字列および16進表示、8進表示文字列の数値化も可能。
 5 STR\$ の逆である。

プログラム例：文字列を、その文字列の表す数値に変換する。

```
10 E$="1.23E-6"
20 A=VAL(E$)*10
30 LPRINT "E$=";E$,"A=";A
40 END
```

実行例

E\$=1.23E-6 A= 1.23E-05

参照：STR\$

APL ・本文と同じ。ただし、&で始まる文字列は扱えない。

TRS1 ・本文と同じ。

CPM ・本文と同じ。

M223 ・APLと同じ。ただし、VAL# (E\$)で高精度数を得ることもできる。

MZK ・APLと同じ。

CBM ・APLと同じ。

PC8 ・本文参照。

TRSII ・本文と同じ。

PC3 ・APLと同じ。ただし、E\$が文字定数の場合、カッコで囲む必要はない。

IF82 ・本文と同じ。

レベル3 ・本文と同じ。

C180 ・本文と同じ。ただし、文字列の先頭が空白だと0になる。
 ・2進数値の符号をなくして10進整数に変換する関数としてFLTがある。逆関数はCHNO%。
 ・FIX%は実数を2進数値に変換する関数。

M243 ・M223と同じ。

MZB ・APLと同じ。

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・APLと同じ。

PC88 ・本文と同じ。

N52 ・E\$の最初の文字が+、-、&、.、空白、数字でなければ0になる。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。

SMC ・本文と同じ。ただし、特徴1、3の場合はエラーとなる。

MZ35 ・本文と同じ。

X1Hu ・本文と同じ。なお、特徴4は2進表示でもよい。

MB16 ・本文と同じ。

IBM55 ・本文と同じ。
 ・この他に、JIS16進表示(&J)、およびJIS区点表示(&K)の文字列も、連続コード番号に変換することができる。

PC100 ・本文と同じ。

MSX ・本文と同じ。

数値の文字列化 STR\$

Eス・ティー・アール\$

形式: STR\$(A)

機能: Aの数値を表す文字列を求める関数。

特徴: 1 Aの数値の大きさにより, 求められるストリングも指数表現のものとなる。
 2 VALの逆である。

プログラム例: 数値を文字列化し, 左から順に1文字ずつ増やしなが
 ら表示する。

```

10 A=123.456
20 E$=STR$(A+111.111)
30 FOR I=1 TO 8
40 LPRINT LEFT$(E$,I)
50 NEXT I
60 END

```

実行例

```

2
23
234
234.
234.5
234.56
234.567

```

参照: VAL

APL ・ 本文と同じ。

TRS₁ ・ 本文と同じ。

CPM ・ 本文と同じ。

M223 NUM\$(A)
 ・ Aには高精度数(A#)も許される。

MZK ・ 本文と同じ。

CBM ・ 本文と同じ。

PC8 ・ 本文参照。

TRS₁₁ ・ 本文と同じ。

PC3 ・ 本文と同じ。

IF8₂ ・ 本文と同じ。レベル₃ ・ 本文と同じ。

C180 ・ 本文と同じ。

M243 ・ M223と同じ。

MZB ・ 本文と同じ。

BUB ・ 本文と同じ。

FM8 ・ 本文と同じ。

PSP STR\$(A, M)
 ・ Mは文字列長で, 有効数字が右づ
 めに変換され, あとはスペースで
 埋められる。Mを省略するとPC8
 と同じ。

PC88 ・ 本文と同じ。

N52 ・ 本文と同じ。

PC6 ・ 本文と同じ。

MLT ・ 本文と同じ。

HC ・ 本文と同じ。

FP11 ・ 本文と同じ。

SMC ・ 本文と同じ。

MZ35 ・ 本文と同じ。

X1Hu ・ 本文と同じ。ただし, 正
 の数は先頭にスペースが入る。

MB16 ・ 本文と同じ。

IBM55 ・ 本文と同じ。

PC100 ・ 本文と同じ。

MSX ・ 本文と同じ。

16進への変換 HEX\$

ヘキサ\$

形式：HEX\$(M)

機能：10進数の引数を、16進数に変えた文字列で与える関数。

- 特徴：
- 1 引数の小数部分を切り捨て、整数に変換されたあとで関数の値が求められる。
 - 2 引数Mの値は、 $-32768 \leq M \leq 32767$ 、または、 $0 \leq M \leq 65535$ の範囲である。
 - 3 16進数で表示するプログラムを作成するときなどに使うと便利である。

プログラム例：-2から3の数を16進文字列に変換する。34も変換する。

```
10 FOR I=-2 TO 3
20 LPRINT I;"=";HEX$(I);" H"
30 NEXT I
40 LPRINT HEX$(34)
50 END
```

実行例

```
-2 =FFFE H
-1 =FFFF H
0 =0 H
1 =1 H
2 =2 H
3 =3 H
34
```

参照：OCT\$, VAL

APL —

TRS₁ —

CPM ・本文と同じ。

M223 —

MZK —

CBM —

PC8 ・本文参照。

TRS₁₁ ・本文と同じ。

PC3 —

IF8₂ ・本文と同じ。

レベル₃ ・本文と同じ。

C180 —

M243 —

MZB —

BUB ・本文と同じ。

FM8 ・本文と同じ。

PSP ・本文と同じ。

PC88 ・本文と同じ。

N52 ・引数の小数部分を四捨五入したあとで変換を行う。

PC6 ・本文と同じ。

MLT ・本文と同じ。

HC ・本文と同じ。

FP11 ・本文と同じ。ただし、 $-32769 < M < 65536$ 。

SMC HEX\$(M, N)

- ・Mの値は $-32768 \leq M \leq 32767$ 。小数点以下は四捨五入される。
- ・Nは16進数に変換したあとに何桁とるかを指定する。4桁まで指定でき、0にすると本文と同じ。
- ・Nを省略すると、0を指定したことになる。

MZ35 ・STRING\$(A1, A2)

- ・A1は桁数、A2に10進で変換する数字を入れる。

X1H_u ・引数は $-65535 \sim 65535$ ($-65535 \sim -1$ は $1 \sim 65535$ と同じ)。

MB16 ・負数では、 $-32768 \sim -1$ で、正数では $0 \sim 65535$ 。

IBM55 ・FP11と同じ。ただし、小数部分は四捨五入される。

PC100 ・本文と同じ。

MSX ・本文と同じ。

8 進数への変換 OCT\$

オクタル\$

形式: OCT\$ (M)

機能: 10進数の引数を, 8進数に変えた文字列で与える関数.

- 特徴: 1 引数の小数部分を切り捨てて, 整数に変換されたあとで関数の値が計算される.
- 2 引数 M の値は, $-32768 \leq M \leq 32767$, または, $0 \leq M \leq 65535$ の範囲である.
- 3 8進数で表示するプログラムを作成するときなどに使うと便利である.

プログラム例: 種々の 10 進数を 8 進文字列に変えて印字する.

```
10 LPRINT OCT$(8), OCT$(34)
20 LPRINT OCT$(-34), OCT$(12.5)
30 END
```

実行例

10	42
177736	14

参照: HEX\$

APL —

TRS₁ ・本文と同じ.

CPM ・本文と同じ.

M223 —

MZK —

CBM —

PC8 ・本文参照.

TRS₁₁ ・本文と同じ.

PC3 —

IF8₂ ・本文と同じ.レベル₃ ・本文と同じ.

C180 —

M243 —

MZB —

BUB ・本文と同じ.

FM8 ・本文と同じ.

PSP —

PC88 ・本文と同じ.

N52 —

PC6 —

MLT ・小数点以下は四捨五入される.

HC ・MLT と同じ.

FP11 ・本文と同じ. ただし,
-32769 < M < 65536.

SMC —

MZ35 —

X1Hu ・引数は -65535 ~ 65535
(-65535 ~ -1 は 1 ~ 65535 と
同じ).
・その他に 2 進数への変換として,
BIN\$(M)がある.

MB16 ・負数は, -32768 ~ -1,
正数は 0 ~ 65535.

IBM55 ・FP11 と同じ. ただし,
小数部分は四捨五入される.

PC100 ・本文と同じ.

MSX ・本文と同じ.
・2 進数への変換として BIN\$(M)
がある.

10進, 60進の変換 CDMS, CDEG

コンバート・デシマル, コンバート・デグリ

形式: ① CDMS (A)

② CDEG (A)

機能: ①10進表現の値Aを, 度・分・秒表現へ変換する.

②度・分・秒表現の値Aを, 10進表現へ変換する.

- 特徴: 1 Aが符号を伴わない数値定数, 数値変数の場合や関数の場合は, カッコで囲む必要はない.
 2 三角関数で単位を DEG に変えた場合, 引数を変換するのに便利である.
 3 DD.MM.SS の形で度・分・秒を表現する.
 4 FIX関数をもつ機種では,
 $DD = \text{FIX}(A)$
 $MM = \text{FIX}((A - DD) * 60)$
 $SS = (A - DD - MM/60) * 3600$
 で代用できる. また CDEG については, $CDEG = DD + MM/60 + SS/3600$.

プログラム例: CDEG, CDMS 命令で, A の値を変換する.

```
10 A=12.45
20 A1=CDEG (A)
30 A2=CDMS (A1)
40 PRINT A1,A2
50 END
```

実行例

12.75

12.45

参照:

APL —

TRS_I —

CPM —

M223 —

MZK —

CBM —

PC8 —

TRS_{II} —

PC3 ・ 本文参照.

IF8₂ —レベル₃ —

C180 —

M243 —

MZB —

BUB —

FM8 —

PSP —

PC88 —

N52 —

PC6 —

MLT —

HC —

FP11 —

SMC —

MZ35 CDMS A
 CDEG A
 ・ 機能は本文と同じ.

X1Hu —

MB16 —

IBM55 —

PC100 —

MSX —

頁欄の太字は、本文で取り上げた、命令・用語解説のページを示す。

あ	オペランド 30 音楽演奏 257	行番号 50, 56 く	再実行 65 最小値 292 最大値 292 削除 57, 83 サーチ 40 座標 229 サブルーチン 66, 81, 105, 118 三角関数 281, 282, 283 算術演算 30 算術演算子 18	条件 91 使用文字 15 剰余 288 常用対数 285 書式 141, 142, 143, 144, 146, 148, 150, 153 シンボル 16
アクセス 279 アクティブ 226 アスキー 20, 301, 323 アセンブラ 296 アセンブリ言語 118 アセンブル 114 アトリビュート 26 アドレス 81, 114 アナログ・ポート 138	か 解除 167 階乗 295 角度 246, 283 格納 161 カセット 200, 202, 204, 206, 207, 208 仮想スクリーン 34 カーソル 34, 217, 219, 220 カーソル座標 247 型変換 30 カタログ 194 可変長 186 可変長レコード 193 紙送り 278 画面 151, 209, 210, 212, 217, 221, 225, 227, 230, 232, 242, 244, 246, 248, 250, 252 仮引数 109 関係演算子 19 関数定義 84	空白 302 クラスタ 191 グラフィック(ス) 15, 151, 225, 227, 240, 244, 247 クリア 49 繰り返し 89, 90, 301 クローズ 64, 174 クロック 306 け ゲーム・パドル 138 行番号 50, 56 こ 孤 236 交換 71, 188 固定小数点型式 23, 322 固定長 175 コード・ワーク 49 コピー 151, 184 コマンド・レベル 50 コミュニケーション・ファイル 192, 199 コミュニケーション・ポート 185, 261 コントロール 323 コントロール文字 34 さ 再開 101	時刻 306 指数 284 自然対数 284, 286 実引数 108 実行 36, 39, 188 シフト 296 シミュレーション 293, 294 修正 48 終端 193 終了 66, 110 出力 139, 150, 152, 153, 154, 157, 158, 177, 179, 190, 322 出力形式 322 ジョイスティック 222 消去 165, 210	す 数値 22, 23, 24, 326, 327 スクリーン・エディタ 32 スクリーン座標 227, 229 スクローリング 214 スクロール 209 スタック 261 ステップ 39 ステートメント 16 ストリング 207, 301, 303 スプライト 226, 251, 255 スロット 54 せ 整数 312, 313, 316, 319 整数化 311 整数型 22, 72, 312 セクタ 189, 190 セクタ番号 196 セグメント 118 絶対値 287 設定 167 セーブ 121, 202, 248, 250, 273 扇形 236 全二重方式 263
い 色 230 印字桁 304 インターフェース 158 インターレース・モード 226	え エクステンション 163 エコー・バック 263 エラー 98, 99, 100, 101, 102, 103 円 236 演算子 19 お 扇形 236 オート・リポート 32 オーバフロー 281 オーバーレイ 44 オープン 46, 172, 259 オプション 36	き キー 267 機械語 86, 112, 119, 121, 123 キャラクタ 233, 248, 250, 252, 254 キャリッジ・リターン 34 境界色 239 行入力 127	シェイプ・テーブル 76 時間 274, 275 式 30 シーケンシャル・ファイル 132, 134, 152, 157, 192, 197, 199 時刻 306 指数 284 自然対数 284, 286 実引数 108 実行 36, 39, 188 シフト 296 シミュレーション 293, 294 修正 48 終端 193 終了 66, 110 出力 139, 150, 152, 153, 154, 157, 158, 177, 179, 190, 322 出力形式 322 ジョイスティック 222 消去 165, 210	条件 91 使用文字 15 剰余 288 常用対数 285 書式 141, 142, 143, 144, 146, 148, 150, 153 シンボル 16

頁欄の太字は、本文で取り上げた、命令・用語解説のページを示す。

線引き 232, 233, 234

そ

相対座標 236

属性 26, 165, 167, 194, 216

ソース・ワーク 49

た

対数 285, 286

代入文 67

タイマ 265, 266

ダイレクト・モード 23

楕円 236

多重 89

タッチ・パッド 223

タッチ・パネル 138

タブ 34

ダミー 116, 184, 277

単精度 23, 73, 314, 315, 317,

320

ダンプ 177

端末機 263

ち

チェック・サム 204

注釈文 63

中断 52

つ

通信回線 261, 262

て

停止 64

ディジタイザ 247

ディスク関数 191

ディスクット 152

ディスプレイ・ページ 226

ディレクトリ 177, 179, 182,

194

テキスト 151, 225

出口 111

データ文 68, 69

デバッグ 28

テープ 206

デュアル・ルーチン 34

テンポ 258

と

統計関数 292

飛び越し 92

トラック 189, 190

トレース文 38

度 281

ドット 240, 242, 244, 247

ドット・グラフィック(ス)

248, 250, 252, 254

ドライブ 36, 189, 190

トリガ・センス 222

トリガ・ボタン 223

な

内部コード 136, 157, 316,

317, 318, 319, 320, 321

ナル・コード 148

ナル・ストリング 129, 298,

299, 303

に

入出力ポート 259

入力 68, 125, 129, 131, 132,

134, 135, 136, 137, 189

ぬ

塗りつぶし 238

ね

ネスティング 89

は

倍精度 24, 74, 314, 315,

318, 321

バイト 137

バイナリ・コード 161

倍々精度型定数 24

配列 29, 78, 80

配列画面出力 252, 254

バック・グラウンド・

カラー 238

バック・グラウンド・

ストリング 239

バック・スペース 34

パッファ 118, 155, 156, 171,

175, 277

パドル 223

パブル・メモリ 279

パリティ 263

パレット 236

半二重方式 263

ひ

引数 137

引き渡し 85

左づめ 155

日付け 308

ビット 296

ビット・パターン 276

表示 216

ふ

ファイル 49, 172, 174

ファイル・ディスクリプタ

189, 190, 194

ファイルの大きさ 199

ファイルの終わり 192

ファイル番号 172

ファイル・ポインタ 186

ファンクション・キー 271,

272, 273

ファンクション・コード

232, 240

フィールド 155, 156, 175

フォア・グラウンド・

カラー 238, 255

フォーマッティング 180

フォーマット 141, 143, 144,

146, 148, 150, 153, 157

フォーム・ファイル 44

副プログラム 44

符号 289

ブザー 256

物理レコード 183

浮動小数点 23, 322

ブリンク 187, 232

プリンタ 149, 179

プリンタ・ヘッド 277

プリント 141, 143, 144,

146, 148

フローチャート 28

プログラマブル・サウンド・

ジェネレータ 257

プロンプト 65

分岐 93, 97, 98, 100, 104,,

105, 262

へ

平方根 280

並列処理 87

ページ 279

ベース・アドレス 77

ベリファイ 204

変更 163, 185

変数 26, 118

変数画面出力 255

ほ

ポインタ 172

ポート 137, 158

ポーレイト 263

ポリウム 188

ま

マウント 169

マージ 42

マルチ・ステートメント 16

頁欄の太字は、本文で取り上げた、命令・用語解説のページを示す。

CDBL	314	CMT	208	CSRV	219	DEFREAL	73	DSKI\$	189
CDEG	31,330	COLOR	230,232,236	CSTOP	208	DEF SEG	77,121	DSKO\$	190
CDMS	31,330	COMBINE	42	CTL	34	DEFSNG	73	E	
CEIL	312	COMMON	41,85	CTR	201,203,204	DEFSTR	75	EDIT	48
CFLASH	232	COM OFF	261	CURSOR	209,217	DEFUSR	86,112	ELSE	91,105
CFIN	315	COM ON	261	CVD	318,321	DEG	281,282,283,330	EMM	173
CFLOAT	315	COM (n) ON	261	CVERIFY	204	DEL	57	END	46,47,66
CGEN	76	COM STOP	261	CVF	325	DELAY	274	END IF	91
CGET	248	COND	84	CVI	316,319	DELETE	28,49,57,165	END SELECT	91
CGPAT\$	76	CONFIG	171	CVS	317,320	DEVF	191	ENT	65
CHAIN	41,44,85	CONNECT	234	D		DEVICE	178	ENVIRON	178
CHANGE DISP	231	CONSOLE	209,212,214,230	DATA	68,69,70	DEVI\$	189	EOF	97,192
CHANNEL	259,264	CONT	53,65	DATE	308	DEVO\$	190	EOR #n	193
CHDIR	178	CONTINUOUS	182	DATE\$	308	DFK	271	EOR%	21
CHKDSK	191	COPY	151,184	DAY	310	DIM	29,78,83,175	EQV	21
CHNO%	326	COPY/P	151	DCLOSE	174	DIR	177,194	ERA	165
CHR\$	271,323	COS	281	DCML	31,314	DIRECT	177,179,191,194	ERASE	83,210
CHARACTER\$	216	CPLOT	246	DE	194	DIRECTORY	177,191	ERL	102
CINT	312	CPOINT	216	DEBUG	38,39	DIR/P	179	ERN	97,103
CIRCLE	236	CPOS	220	DEF	84	DISK	169	ERR	97,101,102,103
CLEAR	81	CPUT	252	DEF ANN	112	DISP	139	ERRL	102
CLI	46,47	CREATE	172,182	DEF ANNO	86	DISP CLEAR	210,231	ERROR	99,100
CLICK	33	CREV	232	DEFCHR\$	76	DISPLAY	234	ESC	52,323
CLIST	179	CRT	173	DEFDBL	74	DISP TIME	308	EVAL	84
CLOAD	121,122,160,200	CSAVE	122,202	DEFFIL	175,185	DISP USING	141	EVENT	275
CLOAD?	204	CSAVE*	202	DEFFILE	135	DLOAD	159	EXCEPTION	100
CLOAD*	200	CSAVE @	58	DEFFIN	73	DOS	47	EXEC	119,167
CLOAD/V	204	CSIZE	246	DEF FLT	73	DRAW	235,237	EXFNO	177
CLOSE	174,206,207	CSNG	315	DEFINT	72	DS	103	EXLINE	102
CLOSE #	97	CSR	139,217	DEFFN	84	DSAVE	161	EXNO	103
CLR	82	CSRCLM	220	DEFFONT	76	DSKF	191	EXP	284
CLS	210	CSRLIN	219	DEF KEY	271,273	DSKINI	180		
CMD	46,47,123								

頁欄の太字は、本文で取り上げた、命令・用語解説のページを示す。

LOAD/T	200	MIN	292	O		OPEN/R	173,206	PEEK	113,114
LOAD/V	160,204	MINS	292			OPEN/S	173,207	PEN	221,222
LOAD?	204	MIRROR	323	OCT \$	329	OPEN#	316,317,318,319,	PEN関数	222
LOC	197,217	MKD \$	155,318,321	ODISP	226		320,321	PENCLN	222
LOCATE	217	MKDIR	178	OFF KEY	267	OPKEY	271,273	PENLIN	222
LOCATES	228	MKF \$	323	OFF TIME	265	OPTION BASE	78,80	PEN OFF	269
LOCK	167	MKI \$	155,316,319	OLD	159	OPTION SCREEN	226	PEN ON	269
LOF	199	MKS \$	155,317,320	ON COM GOSUB	262	OR	21	PEN READ	222,270
LOG ₁₀	285	MLOAD	121	ON COM (n) GOSUB	262	OR%	21	PEN STOP	269
LOG _e	286	MO	169	ONERR	97	OUT	158	PENX	222
LOGIN	61	MOD	288	ON ERROR	97	OUTPUT	172	PENY	222
LOMEM	81	MODIFY	31	ON ERROR GOTO	98,99,	OUTPUT #1	152	PLAY	257,258
LOUT	158	MON	123		100,102	OUTPUT #5 USING	150,	PLOAD	41,44,57
LPOS	219,277	MOTOR	208	ON GOSUB	105		153	PLOT	235,240
LPRINT	16,149,302,304	MOUNT	169,171,173	ON GOTO	93	OVAREA	87	PMAP	229
LPRINT USING	150	MOUSE	224	ON HELP GOSUB	268	P		POINT	244
LPT	173	MOVE	62,67,156	ON INTERVAL	265,266			POKE	113,114
LSET	155,319,320,321	MSAVE	121	ON KEY GOSUB	267	PACF	116	POLLING	264
LTRON	38	MUSIC	256,257,258	ON PEN GOSUB	221,270	PAINT	238	POLY	237
LWIDTH	185,213	N		ON RESTORE	93	PAD	138,223	POP	107
M		NAME	163	ON RESUME	93	PAGE	278	POS	219,220,277,303
MAP 関数	227	NEXT	90	ON RETURN	93	PALET	231	POSITION	255
MARGIN	185,212	NEW	49	ON STRIG	270	PALETTE	231	PRESET	235,240,242
MAX	292	NEW ON	46	ON SPRITE	255	PARACT	87	PRINT 16,139,149,302,304	
MAXFILE (S)	171	NO LIST	58,59	ON STOP GOSUB	268	PAREND	87	PRINT #n	152,186,302
MAXS	292	NORM	232	ON TIME	265	PARSTOP	87	PRINT/T	152
MCLEAR	82	NORMAL	322	ON TIME GOSUB	266	PASS	58,59	PRINT USING	141,143,
MEM	173	NOT	21	OPCHNL	264	PATTERN	255		144,146
MEMSET	81	NOTRACE	38	OPEN	154,206,207	PAUSE	52,274	PRINT #n USING	150,153
MERGE	41,42,161	NULL	129	OPEN ファイル	172	PCOPY	62	PRINT @1, USING	150
MID	300	NUM \$	327	OPEN 入出力	259	PDELETE	57	PRINT/P USING	150
MID \$	300			OPEN/I	173,206	PDL	138,223	PROG	61,84
				OPEN/O	173			PROT	167

頁欄の太字は、本文で取り上げた、命令・用語解説のページを示す。

PRW	226	REM	16, 63, 288	ROUND	290	SET DATE	308	SUB	109
PSET	235, 240	REMOVE	170	ROW	219	SETFORM	214, 234	SUBEND	110, 111
PSG	257	REN	56, 164	RPOS	197	SETQ	84	SUBEXIT	111
PURGE	165	RENAME	163, 167	RS-232C	259, 263	SET TIME	306	SWAPVALUE	44, 71, 108
PUT 135, 154, 197, 199, 252		RENUM	56, 58	RSET	156, 319, 320, 321	SFUNC	271	SWAPプログラム	188
PUT SPRITE	253	RENUMBER	56	RUN	36	SGN	289	SYMBOL	246
PUT @	252	REPLACE \$	300	RV	231	SHELL	46	SYS	119, 123
PUT @A	254	REPEAT	33, 96	S		SIGNAL	87	SYSGEN	180
PUT# 152, 186, 319, 320, 321		REP \$	301, 302			SIN	281	SYSTEM	46, 47, 61, 124
Q		REQ	259	SAVE	122, 159, 161, 202, 207	SIZE	116	T	
QUAD	235	RES	70	SAVEM	121, 122	SKIP	200	TAB	125, 139, 148, 304
R		RESET	123, 169, 170, 174, 242	SAVE/P	53, 58	SKIPF	208	TABLE	233
RAD	281, 282, 283	RESIDENT	42	SAVE/T	202	SLEEP	274	TAN	281
RAM ファイル	175	RESTORE	70, 182, 186	SBIT%	21	SL%	21, 296	TAPCNT	208
RANDOM	294	RESUME	101	SCALE	228	SOUND	256, 257	TELCOM	264
RANDOMIZE	294	RETRY	101	SCOPY	184	SPC	302	TEMPO	256, 257, 258
RBIT%	21	RETURN	52, 101, 106	SCR	173	SPRITE	251, 255	TERM	263
RCV	264	REV	232	SCRATCH	165, 172	SPACE	52, 302	THEN	91
RCY	219	REW	208	SCREEN関数	216	SQR	280	TID%	259
RE	170	RFORMAT	175, 316, 317, 318, 319, 320, 321	SCREENモード	225	SR%	21, 296	TIM	306
READ	68, 70, 135, 167	RIGHT	298	SCRN	173, 216	ST%	259	TIME	306
READ #n	136	RJUST	156, 176	SCRN \$	216	START	87	TIME OFF	265
REBL	231	RLIST	264	SCROL	214	STAT	61, 116, 191	TIME ON	265
REC	197	RMDIR	178	SCROLL	214	STATUS	116	TIMEOUT	275
RECDEF	175	RNAME	164	SDISP	226	STEP	39	TIMER	265, 306
RECEIVE	136	RND	293	SEARCH	40, 303	STOP	52, 64	TIME STOP	265
RECORD	135, 154, 156, 175, 186	ROLL	214	SECT	91	STOP ON	268	TIME \$	266, 306, 308
RECORDSIZ	155, 156, 175, 316, 317, 318, 319, 320, 321	ROM カートリッジ	199	SEND	136, 157, 264	STORE	161, 162	TI \$	306
RELEASE	170	ROM パック	178, 181	SENREV	264	STR \$	327	TITLE	61
		ROPEN	206	SEP	193	STRIG	222, 270	TRACE	38
		ROPEN/T	206	SET	67, 165, 167, 194, 240	STRINGS \$	301	TRANSFER	87
				SETATR	167	STRPTR	118	TRAP	97, 100

頁欄の太字は、本文で取り上げた、命令・用語解説のページを示す。

TROFF	38	W	
TRON	38		
TR PRINT	38	WAIT	87, 274
TYPE	139, 178	WAIT時間	275
U		WAITデータ	276
UDIM	80	WEND	95
UN LIST	58	WHILE	95
UNLOCK	167	WID	125, 139
UNTIL	96	WIDTH	185, 212
UPD	167	WIDTH LPRINT	185
URGENCY	87	WIDTH USING	213
USR	86, 112	WIND	208
V		WINDOW	227
VAL	326, 327	WIPE	211
VARLIST	118	WOPEN	172, 207
VARPTR	118	WOPEN/T	207
VDP	113, 115	WRITE	139, 154, 167
VERIFY	167, 204	WRITE #n	136, 157
VIDEO-RAM	187	WRITE #5 USING	153
VIEW	227	X	
VINIT	180	XOR	21
VINP	113		
VLOAD	187		
VOUT	114		
VPEEK	113		
VPOKE	114		
V RAM	113		
VSAVE	187		
VTAB	217		
VTCLEAR	83		

10進表示=上位の10進数+下位の10進数

JIS 8単位符号表
(00~7F までは、
ASCIIコードと同じ)

上位 下位	0		16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	0	NUL	DLE		0	@	P	`	p	機 能 符 号	機 能 符 号		一	タ	ミ	国 字 符 部 分	国 字 符 部 分
1	1	SOH	DC1	!	1	A	Q	a	q			。	ア	チ	ム		
2	2	STX	DC2	"	2	B	R	b	r			「	イ	ツ	メ		
3	3	ETX	DC3	#	3	C	S	c	s			」	ウ	テ	モ		
4	4	EOT	DC4	\$	4	D	T	d	t			、	エ	ト	ヤ		
5	5	ENQ	NAK	%	5	E	U	e	u			.	オ	ナ	ユ		
6	6	ACK	SYN	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7	7	BEL	ETB	'	7	G	W	g	w			ァ	キ	ヌ	ラ		
8	8	BS	CAN	(8	H	X	h	x			ィ	ク	ネ	リ		
9	9	HT	EM)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
10	A	LF	SUB	*	:	J	Z	j	z			ェ	コ	ハ	レ		
11	B	VT	ESC	+	;	K	[k	{			ォ	サ	ヒ	ロ		
12	C	FF	FS	,	<	L	¥	l				ャ	シ	フ	ワ		
13	D	CR	GS	—	=	M]	m	}			ュ	ス	ヘ	ン		
14	E	SO	RS	.	>	N	^	n	—			ョ	セ	ホ	ッ		
15	F	SI	US	/	?	O	—	o	DEL			ツ	ソ	マ	°		

機能キャラクタの定義

NUL	空 白	DLE	伝送制御拡張
SOH	ヘディング開始	DC ₁	装置制御 1
STX	テキスト開始	DC ₂	装置制御 2
ETX	テキスト終結	DC ₃	装置制御 3
EOT	伝送終了	DC ₄	装置制御 4
ENQ	問い合わせ	NAK	否定応答
ACK	肯定応答	SYN	同期信号
BEL	ベル, 注意を喚起する	ETB	伝送ブロック終結
BS	後 退	CAN	取り消し
HT	水平タブ	EM	媒体終端
LF	改 行	SUB	置換キャラクタ
VT	垂直タブ	ESC	拡 張
FF	書式送り	FS	ファイル分離キャラクタ
CR	復 帰	GS	グループ分離キャラクタ
SO	シフト・アウト	RS	レコード分離キャラクタ
SI	シフト・イン	US	ユニット分離キャラクタ

☒ 誘導関数

BASICに用意されていない関数のうち、いくつかは、次のように導き出すことができる。

○常用対数

$$\text{LOG}_{10}X = \text{LOG}_e(X) / \text{LOG}_e(10)$$

○セカント

$$\text{SEC}(X) = 1 / \text{COS}(X)$$

○コセカント

$$\text{COSEC}(X) = 1 / \text{SIN}(X)$$

○コタンジェント

$$\text{COT}(X) = 1 / \text{TAN}(X)$$

○アークサイン

$$\text{ARCSIN}(X) = \text{ATN}(X) / \text{SQR}(1 - X * X)$$

○アークコサイン

$$\text{ARCCOS}(X) = -\text{ATN}(X / \text{SQR}(1 - X * X)) + \pi / 2$$

○アークセカント

$$\text{ARCSEC}(X) = \text{ATN}(\text{SQR}(X * X - 1)) + (\text{SGN}(X) - 1) * \pi / 2$$

○アークコセカント

$$\text{ARCOSEC}(X) = \text{ATN}(1 / \text{SQR}(X * X - 1)) + (\text{SGN}(X) - 1) * \pi / 2$$

○アークコタンジェント

$$\text{ARCCOT}(X) = -\text{ATN}(X) + \pi / 2$$

○ハイパーボリック・サイン

$$\text{SINH}(X) = (\text{EXP}(X) - \text{EXP}(-X)) / 2$$

○ハイパーボリック・コサイン

$$\text{COSH}(X) = (\text{EXP}(X) + \text{EXP}(-X)) / 2$$

○ハイパーボリック・タンジェント

$$\text{TANH}(X) = (\text{EXP}(2 * X) - 1) / (\text{EXP}(2 * X) + 1)$$

○ハイパーボリック・セカント

$$\text{SECH}(X) = 2 / (\text{EXP}(X) + \text{EXP}(-X))$$

○ハイパーボリック・コセカント

$$\text{COSECH}(X) = 2 / (\text{EXP}(X) - \text{EXP}(-X))$$

○ハイパーボリック・コタンジェント

$$\text{COTH}(X) = (\text{EXP}(2 * X) + 1) / (\text{EXP}(2 * X) - 1)$$

○エリア・ハイパーボリック・サイン (SINHの逆関数)

$$\text{ARSINH}(X) = \text{LOG}_e(X + \text{SQR}(X * X + 1))$$

○エリア・ハイパーボリック・コサイン (COSHの逆関数)

$$\text{ARCOSH}(X) = \text{LOG}_e(X + \text{SQR}(X * X - 1))$$

○エリア・ハイパーボリック・タンジェント (TANHの逆関数)

$$\text{ARTANH}(X) = \text{LOG}_e((1 + X) / (1 - X)) / 2$$

○エリア・ハイパーボリック・セカント (SECHの逆関数)

$$\text{ARSECH}(X) = \text{LOG}_e((\text{SQR}(1 - X * X) + 1) / X)$$

○エリア・ハイパーボリック・コセカント (COSECHの逆関数)

$$\text{ARCOSECH}(X) = \text{LOG}_e((\text{SGN}(X) * \text{SQR}(1 + X * X) + 1) / X)$$

○エリア・ハイパーボリック・コタンジェント (COTHの逆関数)

$$\text{ARCOTH}(X) = \text{LOG}_e((X + 1) / (X - 1)) / 2$$

(注) $\pi \approx 3.14159265$ (円周率)

1984年版

最新マイコンBASIC規格表

昭和57年6月1日 初版発行
昭和59年6月1日 発行
S. 59. 6. 1 第1刷

©1982

編著者 中村八束
編集発行人 飛坐博
発行所 CQ出版株式会社
〒170 東京都豊島区巣鴨1-14-2

定価 900円

電話 (03)947-6311(代表)
振替 東京0-10665

印刷・製本 関印刷(株)

■ マイコンBASIC・メーカー一覧

- ▷ APPLE II / IIe : ▷キヤノン販売(株) 情報機器営業部 ☎03(455)9223
〒108 東京都港区三田3-11-28
▷(株)イーエスディラボラトリ ☎03(816)3911
〒113 東京都文京区湯島4-1-11
- ▷ TRS I / II / III : (株)エー・アンド・エー・ジャパン タンディ ラジオ シャック事業部 ☎0424(88)3500
〒182 東京都調布市多摩川1-44-1
- ▷ M223/243/23/20 : ソード(株) ☎03(281)8111
〒104 東京都中央区八重洲2-7-12京橋K-1ビル
- ▷ MZ80K/C/B : シャープ(株) ☎06(621)1221
PC3100/3200/3500 X1 〒545 大阪府大阪市阿倍野区長池町22-22
- ▷ CBM4000 : コモドール・ジャパン(株) ☎03(433)6111
〒105 東京都港区浜松町1-1-11
- ▷ PC8001/8001^{MKII} : 日本電気(株) NECパソコン インフォメーション センター ☎03(452)8000
PC8800/9800 〒108 東京都港区三田3-14-10
PC100
- ▷ IF800 : 沖電気工業(株) OA事業部営業第3部 ☎03(454)4017
〒108 東京都港区芝浦4-10-3

- ▷ ベーシックマスター : 日立家電販売(株) ☎03(502)2111
レベル3/MB16000 〒105 東京都港区西新橋2-15-12
- ▷ C180/C18/F9450 : パナファコム(株) ☎0462(64)3351
FM X 〒242 神奈川県大和市深見534
- ▷ FM8/7 : 富士通(株) 半導体統轄営業部 ☎03(502)0161
〒105 東京都港区虎ノ門2-3-13
- ▷ PASOPIA : 東京芝浦電気(株) オフィスオートメーション事業部 ☎03(567)6758
〒105 東京都港区虎ノ門1-26-5
- ▷ MULTI 16 : 三菱電機(株) パーソナルコンピュータ部 ☎03(218)3543
〒100 東京都千代田区丸の内2-2-3
- ▷ HC20 : エプソン(株) ☎02635(4)0271
〒399-07 長野県塩尻市広丘原新田80
- ▷ FP1100/1000 : カシオ計算機(株) ☎03(347)4811
〒160 東京都新宿区西新宿2-6-1
- ▷ SMC70/HB55 : ソニー(株) お客様相談センター ☎03(448)3311
〒141 東京都大崎局区内ソニー(株)
- ▷ IBM 5550 : 日本アイ・ビー・エム(株) ☎03(586)1111
〒106 東京都港区六本木3-2-12

TOSHIBA

学習にも、ビジネスにも。 システムアップ自由自在。

使い方に合わせて、思い通りのシステムが組めるパソピア7。ずらりとそろった周辺機器。パソピア7を核に、さあ、どんなコンピュータワールドが広がるのか。思うままにシステムを組む。学習に、ゲームに、オリジナルプログラム作成に、ますます創造力を刺激するパソピア7。もちろんビジネスにも即対応。汎用簡易言語や日本語ワープロとしてだけでなく、CP/Mにより、高級言語や、さらに豊富なソフトウェアの活用も可能。拡張性でも差をつけたパソピア7。個性の数だけシステムが組める。

SYSTEM 1 OA・ビジネス用途として、CP/Mも使える

フラインカーディスプレイ
PA7165 98,000円

パソピア7 PA7007 119,800円

ミニフロッピー
ディスクユニット
PA7221 159,000円



漢字ROM PAC2
PA7247 40,000円

ドットプリンターII
PA7253 139,000円

システム価格 514,800円(446,600円)

SYSTEM 2 音声合成を楽しむなら

フラインカーディスプレイ
PA7165 98,000円

パソピア7 PA7007 119,800円

ボイスユニット
PA7381 29,800円

コンパクトフロッピー
ディスクユニット
PA7232 69,800円



システム価格 317,400円(249,200円)

- ()内のシステム価格は、グリーンディスプレイ化の組合わせの場合です。
- ※印の商品は、システム価格に含まれません。●日本語ワープロを使用する場合、漢字ROM PAC2が必要です。

SOUND & GRAPHICS 東芝パーソナルコンピュータ

使い方360度 PASOPIA 7

●資料請求は、資料請求券を貼り、住所・氏名・年齢・職業を明記し、〒105東京都港区芝浦1-1-1 株式会社 東芝 OA機器事業部 (03) 457-2951までお申し込みください。●パソピア7を実際にお試しになりたい方は、お近くの東芝パソコンサロン札幌 (011) 221-5023/仙台 (0222) 67-5018/大宮 (0486) 51-1100/秋葉原 (03) 255-0901/銀座 (03) 574-0941/渋谷 (03) 499-5571/名古屋 (052) 202-1048/大阪 (06) 344-0765/広島 (082) 249-6762/福岡 (092) 711-1915/パソピア富山 (0764) 91-2877まで、どうぞ。

資料請求券
PASOPIA7
BASIC言語規格

先端技術をくらしの中に… E&Eの東芝



最新
子
32B
AS
C
組規格表

80

CQ出版